Steuerungsintegrierte, adaptive Programmausführung einer aufgabenorientierten Programmierung in STEP-NC

Von der Fakultät für Maschinenwesen
der Rheinisch-Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von
Diplom-Ingenieur Jochen Wolf
aus
Bergheim/Erft

Berichter:

Universitätsprofessor Dr.-Ing. Christian Brecher
Universitätsprofessor em. Dr.-Ing. Dr.-Ing. E.h. Dr.-Ing. E.h. Manfred Weck

Tag der mündlichen Prüfung: 9. März 2009

Diese Dissertation ist auf den Internetseiten der Hochschulbibliothek online verfügbar.

ERGEBNISSE AUS DER PRODUKTIONSTECHNIK

Jochen Wolf

Steuerungsintegrierte, adaptive Programmausführung einer aufgabenorientierten Programmierung in STEP-NC

Herausgeber:
Prof. Dr.-Ing. Dr.-Ing.E.h. F. Klocke
Prof. Dr.-Ing. Dipl.-Wirt.Ing. G. Schuh
Prof. Dr.-Ing. C. Brecher
Prof. Dr.-Ing. R. Schmitt

Band 2/2009

Fraunhofer Institut Produktionstechnologie

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Autor: Jochen Wolf

Steuerungsintegrierte, adaptive Programmausführung einer aufgabenorientierten Programmierung in STEP-NC

Gedruckt auf holz- und säurefreiem Papier, 100% chlorfrei gebleicht.

© Apprimus Verlag, Aachen, 2009

Wissenschaftsverlag des Instituts für Industriekommunikation und Fachmedien an der RWTH Aachen

Steinbachstr. 25, 52074 Aachen

Internet: www.apprimus-verlag.de, E-Mail: info@apprimus-verlag.de

Die Verwertung der Texte und Bilder, auch auszugsweise, ist ohne Zustimmung des Verlags urheberrechtswidrig und strafbar. Dies gilt auch für Vervielfältigungen, Übersetzungen, Mikroverfilmung und für die Verarbeitung mit elektronischen Systemen.

Printed in Germany

ISBN 978-3-940565-28-0

D 82 (Diss. RWTH Aachen University, 2009)

Diese Dissertation ist auf den Internetseiten der Hochschulbibliothek online verfügbar.

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Werkzeugmaschinenlabor (WZL) der Rheinisch-Westfälischen Technischen Hochschule Aachen.

An vorderster Stelle bedanke ich mich bei Herrn Prof. em. Dr.-Ing. Dr.-Ing. E.h. Dr.-Ing. E.h. Manfred Weck. Er ermöglichte es mir an seinem ehemaligen Lehrstuhl in vielfältigen Projekten und in einem fachlich breit aufgestellten Kollegium ein umfangreiches Wissen in der Automatisierungstechnik zu erlernen und eigenverantwortlich in internationalen Projekten einzusetzen. Für das mir dabei entgegengebrachte Vertrauen, die Freiräume sowie die andauernde Förderung und Diskussion meiner Tätigkeit bedanke ich mich sehr herzlich. Ebenso danke ich Herrn Professor Weck für die Übernahme des Koreferats.

Herrn Prof. Dr.-Ing. Christian Brecher, dem jetzigen Inhaber des Lehrstuhls für Werkzeugmaschinen, danke ich für die Fortsetzung der guten Betreuung, die kritische Durchsicht und Diskussion meiner Arbeit sowie die Übernahme des Referats.

Meinen ehemaligen Kollegen am WZL, insbesondere den Herren Dr.-Ing. Andreas Kahmen, Dr.-Ing. Henning Schulze-Lauen, Dr.-Ing. Claus Hardebusch und Dr.-Ing. Jens Brühl danke ich für die Einführung in die Softwareentwicklung sowie die späteren gemeinsamen Arbeiten rund um die Programmierschnittstelle *STEP-NC* und die institutseigene NC-Steuerung *WZL-NC*. Ebenso danke ich meinen Projektpartnern im Sonderforschungsbereich *Autonome Produktionszelle* und in den internationalen *STEP-NC* Projekten für die gute Zusammenarbeit.

Für die Unterstützung bei der Realisierung und Evaluation bedanke ich mich bei meinen ehemaligen studentischen Hilfskräften, den Herren Kamil Fayzullin, Martin Kunc und Jörg Wiegelmann. Besonders hervorheben möchte ich Herrn Fayzullin, ohne dessen Programmierkenntnisse und dessen ausdauernden Einsatz die Implementierung der vorliegenden Arbeit nicht so umfassend möglich gewesen wäre.

Zudem danke ich meinen ehemaligen Institutskollegen Dr.-Ing. Andreas Kahmen und Dr.-Ing. Roman Klement für die vielen wertvollen Diskussionen und die kritische Durchsicht des Manuskripts.

Ebenso danke ich meinen Eltern, die mir meine Ausbildung ermöglicht haben und die mich stets mit großem Engagement gefördert haben.

Mein größter Dank gilt jedoch meiner lieben Frau Kerstin. Sie hat mir geduldig alle Freiheiten gelassen, die mir nötig waren, um diese Dissertation zu erarbeiten und abzuschließen. Ihr Rückhalt und ihre fortwährende Motivation haben mir sehr geholfen. Herzlichen Dank!

Köln im März 2009 Jochen Wolf

Inhalt

1. Einleitung ... 1

2. Stand der Technik ... 5

2.1. NC-Programmierung ... 5
2.1.1. Verfahren zur NC-Programmierung ... 6
2.1.2. NC-Programmierschnittstelle ... 11
2.1.3. Einfahren und Korrektur von NC-Programmen ... 13
2.2. NC-Steuerungen und Programmverarbeitung ... 14
2.2.1. Ablauf der NC-Programmverarbeitung ... 14
2.2.2. Aufbau einer NC-Steuerung ... 17
2.2.3. Offene NC-Steuerungen ... 18
2.3. Prozessüberwachung und –regelung ... 22
2.3.1. Anwendungen zur Prozessregelung ... 22
2.3.2. Entwicklungsansätze in der Prozessüberwachung und –regelung ... 25
2.3.3. Steuerungsintegration von Überwachungs- und Regelungssystemen ... 29
2.4. Fazit ... 30

3. Zielsetzung und Vorgehensweise ... 33

3.1. Zielsetzung ... 33
3.2. Vorgehensweise ... 34

4. Anforderungen an die Programmierung und Programmverarbeitung ... 37

4.1. Anforderungen an das NC-Programm ... 37
4.2. Anforderungen an die NC-Steuerung ... 39
4.3. Anforderungen an die Benutzerschnittstelle ... 40

5. Konzept einer aufgaben- und merkmalsorientierten Prozessregelung ... 41

5.1. Aufgaben- und prozessbeschreibende NC-Programmierschnittstelle ... 43
5.1.1. Formaler Aufbau der Programmierschnittstelle ... 43
5.1.2. Datenelemente der NC-Programmierschnittstelle ... 46
5.2. Prozessverfolgung und -eingriffe über NC-Programmierobjekte ... 51
5.2.1. Aufgaben- und prozessorientierte Fortschrittsverfolgung ... 52
5.2.2. Aufgaben- und prozessorientierte Bearbeitungseingriffe ... 53
5.3. Modular erweiterbare Software-Architektur ... 55
5.3.1. Kommunikationsschnittstelle auf der Basis des NC-Programms ... 56
5.3.2. Durchgängige Systemerweiterung über die NC-Programmierschnittstelle ... 57
5.3.3. Aufgaben- und merkmalsorientierte Erweiterung der Funktionsmodule ... 58

6. Steuerungskern zur adaptiven Programmverarbeitung ... 61

6.1. NC-Workingsteps zur steuerungsinternen Programmverarbeitung ... 63
6.1.1. Verknüpfte, interaktive NC-Workingsteps ... 63
6.1.2. Ablauf-, Aufgaben- und Funktionsbeschreibung ... 66
6.1.3. Softwaretechnisches Konzept ... 67
6.2. Modular erweiterbare Programminterpretation ... 71
6.2.1. Einlesen des NC-Programms und Zugriff auf dessen Daten ... 72
6.2.2. Abbilden des NC-Programms auf NC-Workingsteps ... 74
6.2.3. Zeitoptimierung durch parallele Vorverarbeitung ... 77

6.2.4. Fortschrittsteuerung auf Basis der NC-Workingsteps ... 79
6.3. Steuerungsinterne Kommunikation der NC-Workingsteps ... 79
6.3.1. Kommunikationskonzepte ... 79
6.3.2. Nachführen des Bearbeitungsfortschritts ... 82
6.3.3. Schnittstellen zur Kommunikation von Eingriffen ... 83
6.4. Weg- und funktionssynchrone Umsetzung der NC-Workingsteps ... 85
6.4.1. Geometriekette ... 85
6.4.2. Synchronisation der Eingriffe zwischen NC und SPS ... 89
6.5. Umsetzung der Eingriffe und Übergangsstrategien ... 91
6.5.1. Änderung der programmierten Arbeitsschrittreihenfolge ... 91
6.5.2. Vorausschauende Prozessoptimierung ... 92
6.5.3. Direkte Eingriffe in die Sollwerte ... 94
6.5.4. Eingriffe in die Bahnverarbeitung ... 96

7. Benutzerschnittstelle zur NC-Programmierung und Manipulation ... 101

7.1. Grundsystem für einen modularen Aufbau ... 102
7.1.1. Hardware und Betriebssystem ... 103
7.1.2. Anwendergerechte, kontextorientierte Interaktion ... 103
7.1.3. Modular erweiterbare HMI-Funktionen ... 106
7.1.4. Das NC-Programm als einheitliche Datenbasis ... 109
7.1.5. Nachrichtensystem zum Datenaustausch und Funktionsaufruf ... 113
7.1.6. Schnittstellenserver zur Kopplung mit der NC-Steuerung ... 113
7.2. Aufgabenorientierte Arbeitsweise ... 116
7.2.1. NC-Programmierung an der NC-Steuerung ... 116
7.2.2. Simulation und Bearbeitungsverfolgung ... 118
7.2.3. Prozessüberwachung und –regelung ... 121

8. Realisierung einer NC-programmbasierten Prozessregelung ... 123

8.1. Aufbau und Teilfunktionen der realisierten Prototypen ... 123
8.1.1. Zugriff auf die NC-Programmierschnittstelle ... 124
8.1.2. CAM-Systeme zur ISO 14649 Programmerstellung ... 125
8.1.3. Steuerungsanbindung über WOP-Systeme ... 126
8.1.4. Statische NC-Programmumsetzung im Interpreter ... 127
8.1.5. Regelbare NC-Programmumsetzung in der NC-Steuerung ... 128
8.2. Adaptive Prozessregelung am Beispiel der Taschenbearbeitung ... 131
8.2.1. Adaption der Schnitttiefe ... 133
8.2.2. Adaption der Werkzeugüberdeckung ... 133
8.3. Praktische Anwendung und Evaluation ... 135
8.3.1. Austauschbarkeit der Aufgaben-, Merkmals- und Operationsdaten ... 135
8.3.2. Zeitverhalten in einer modularen, dezentralen Steuerungsarchitektur ... 135

9. Zusammenfassung und Ausblick ... 139

9.1. Durchgängige, aufgabenorientierte NC-Programmierschnittstelle ... 139
9.2. Autonome, adaptierbare Programmverarbeitung in der NC-Steuerung ... 140

10. Literaturverzeichnis ... 145

11. Anhang ... I

11.1. Auszüge aus der entwickelten NC-Programmierschnittstelle ISO 14649 ... I
11.1.1. Projekt, Arbeitsplan, Arbeitsschritt und NC-Funktionen ... I
11.1.2. Arbeitsschritt zur Bearbeitung eines Features mit einer Operation ... II

11.1.3. Geometrie- und Bearbeitungsmerkmale..... III
11.2. Beispielhafte Erstellung eines ISO 14649 NC-Programms IV
11.3. ISO 14649 Beispielprogramm des STEP-NC Projektkonsortiums..... VIII

Abkürzungen

Abbreviations

ACC	Adaptive Control with Constraints (DIN 66245)
ACO	Adaptive Control for Optimisation
API	Application Programming Interface
APT	Automatically Programmed Tools
APZ	Autonome Produktionszelle des SFB 368
BAPSI	Brankamp-Artis-Prometec-Standard-Interface
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CAM	Computer Aided Manufacturing
CAP	Computer Aided Planning
CLDATA	Cutter Location Data (DIN 66215 / ISO 3592)
CNC	Computerised Numerical Control
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
DCOM	Distributed Component Object Model
DLL	Dynamic Link Library
DIN	Deutsches Institut für Normung
EDM	Electrical Discharge Machining
FFT	Fast Fourier Transformation
FiFo	First in first out
HMI	Human Machine Interface
ISO	International Organisation for Standardisation
JOG	Modus Handbetrieb
MDA	Automatikmodus
MMC	Man Machine Control
NC	Numerical Control, hier gleichbedeutend mit CNC genutzt
NCK	NC Kern
NCS	NC-Schnittstelle / Datenschnittstelle der Bosch Typ 3 osa
OEM	Original Equipment Manufacturer
OLE	Object Linking and Embedding
OMAC	Open Modular Architecture Controller
OPC	OLE for Process Control
OPI	Original Equipment Manufacturer’s Programmer’s Interface
ORB	Object Request Broker

OSACA	Open System Architecture for Controls within Automation Systems
REF	Modus zur Referenzieren der Maschine
SBL	Modus Einzelsatz im Automatikbetrieb
SERCOS	Serial Realtime Communication System (IEC 61491, EN61491)
SFB	Sonderforschungsbereich der Deutschen Forschungsgesellschaft
SPS	Speicherprogrammierbare Steuerung, Anpassteuerung
STEP	Product data representation and exchange (ISO 10303)
STEP-NC	Synonym für die ISO 14649 NC-Programmierschnittstelle und die damit verbundenen Forschungsprojekte
WOP	Werkstattorientierte NC-Programmierung
WZL	Laboratorium für Werkzeugmaschinen und Betriebslehre der RWTH Aachen
WZL-NC	Am WZL entwickelte, verteilte, offene, OSACA konforme NC-Steuerung für die fünfachsige Fräs- und Laserbearbeitung
XML	Extensible Markup Language

1. Einleitung

Introduction

Numerisch gesteuerte Werkzeugmaschinen haben sich in der Produktion etabliert. Bei den in Deutschland hergestellten, spanenden und abtragenden Werkzeugmaschinen liegt ihr Anteil bei über 85% [VDW04]. Gegenüber der Fertigung mit manuell oder mechanisch gesteuerten Maschinen bieten sie eine Vielzahl von Vorteilen. Hierzu zählen die Automation von Bearbeitungsabläufen, kurze Durchlaufzeiten, eine hohe Reproduzierbarkeit bei gleichbleibender Qualität und die flexible Bearbeitung unterschiedlicher Bauteile [WECK01c, KIEF05].

Die Ausnutzung dieser Potentiale hängt jedoch nicht nur von der Numerischen Steuerung (NC) und der anzusteuernden Werkzeugmaschine ab. In einem hohen Maße bestimmen das Steuerungsprogramm, in dem der Bearbeitungsablauf ausgehend von einem idealisierten Prozessmodell im Vorfeld der Bearbeitung kodiert wurde, und seine Anpassung an die reale Fertigungssituation das Ergebnis und die Wirtschaftlichkeit der NC-Bearbeitung [EVER94]. Fortlaufend fordert die Industrie daher zum einen unterstützende Systeme zur schnellen Bereitstellung fehlerfreier und optimierter NC-Programme [BITZ96, STEP99, NCG05] und zum anderen Regelsysteme, um bei Prozessabweichungen automatisiert oder manuell in den laufenden Bearbeitungsprozess eingreifen zu können [VDW02].

Ausgehend von diesen Forderungen werden seit der Entwicklung der ersten NC-Steuerung rechner- und computergestützte Verfahren zur Erstellung möglichst fehlerfreier und optimierter NC-Programme entwickelt. In Abhängigkeit von dem Einsatzort und der Komplexität der Bearbeitungsaufgabe stehen der Industrie heute verschiedenartige Planungs-, Programmier- und Simulationssysteme zur Verfügung. Aus Gründen der Systemkosten, der Arbeitsumgebung und der Mitarbeiterqualifikation nimmt der Funktionsumfang dieser Systeme ausgehend von der Arbeitsvorbereitung über die Werkstattprogrammierung bis hin zu den in die Steuerungsoberfläche der Maschine integrierten Programmierhilfen stetig ab. Daraus ergibt sich insbesondere an der Werkzeugmaschine eine suboptimale Unterstützung bei der Erstellung, Erprobung und Korrektur von NC-Programmen [NCG93, ENGS02]. In der Folge werden die Potentiale von Prozess, Maschine und Werkzeug nicht ausgenutzt. Zudem führen nicht erkannte Programmierfehler zu Qualitätseinbußen am Bauteil oder aufgrund von Kollisionen zu Schäden.

Die Aufgabe, das NC-Programm an der Maschine zu erproben und an die realen Fertigungsbedingungen anzupassen, übernehmen mehrheitlich die Werker an der Maschine und nicht speziell ausgebildete Einrichter oder Meister [LAYG05]. Dabei werden an den Werker besondere Anforderungen gestellt. Er muss die automatisiert ablaufenden, schnellen, komplexen und oftmals schwer einsehbaren Bearbeitungsprozesse überwachen, Abweichungen sicher erkennen und rechtzeitig geeignete Gegenmaßnahmen bestimmen und einleiten [SCHU00].

Bei dieser komplexen Tätigkeit wird der Werker nur unzureichend unterstützt. Anstelle einer werkergerechten, seinem Prozessverständnis entsprechenden Aufgabenbeschreibung, muss er die Bearbeitung in Form von steuerungsorientierten Anweisungen programmieren und optimieren [SCHI95, BROU00]. Den Abarbeitungsfortschritt zeigt die Steuerung zeilenweise in Form der im NC-Programm kodierten Befehle und Parameter an. Übergeordnete Operations- und Strategiebeschreibungen fehlen ebenso, wie eine grafische Darstellung der Fertigungsgeometrie, der zu verfahrenden Werkzeugbewegungen und deren Auswirkung. Zudem arbeitet die NC-Steuerung „nach Programm“. D.h. sie sperrt während der Bearbeitung den Zugriff auf den im NC-Programm vorgegebenen Arbeitsplan, die Technologieparameter, die von den Werkzeugen abhängige Bahnauflösung, Weg- und Schaltbefehle. Zur Korrektur des Bearbeitungsprozesses kann nur auf wenige, global wirkende Steuerungsparameter zugegriffen werden, Bild 1.1. Im Fall der Fräsbearbeitung sind die Eingriffsmöglichkeiten auf den Bahnvor-

schub, die Werkzeugdrehzahl und, je nach Steuerungshersteller, auf NC-Variablen beschränkt.

Um dagegen die Operationsfolge, die Werkzeuge oder die Bahnstrategie ändern zu können, muss die Bearbeitung unterbrochen werden. Die fehlerhaften oder suboptimalen Programmvorgaben werden an der Maschine oder in der Arbeitsvorbereitung korrigiert. Danach wird das NC-Programm erneut in die Steuerung geladen und die Abarbeitung gestartet. Ein bearbeitungsparalleler Zugriff auf die Programmvorgaben ist nicht möglich. Durch die mitunter iterativen Korrekturschleifen geht wertvolle Produktionszeit verloren. Zudem lassen sich während der offline erfolgenden Korrektur des Programms Fehleingaben und ungeeignete Prozessvorgaben nicht ausschließen.

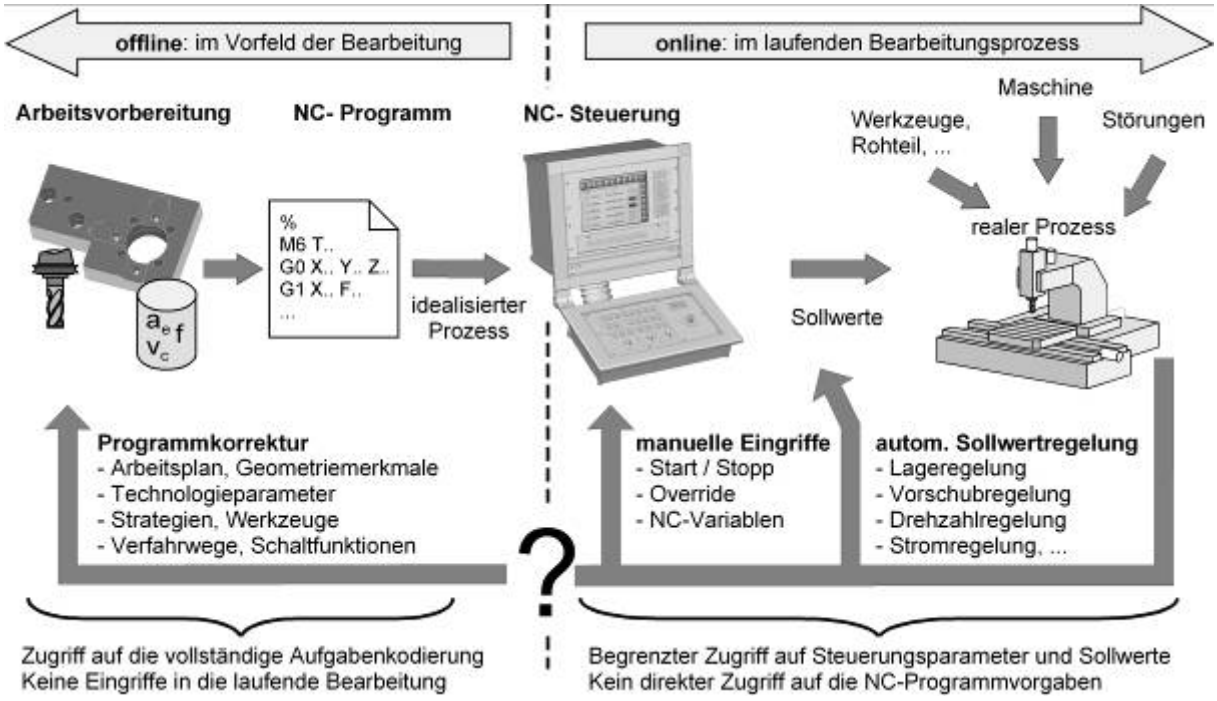


Bild 1.1: Fehlende Durchgängigkeit beim Zugriff auf die NC-Programmvorgaben
Lack of consistency while accessing NC commands and set values

Vor dem Hintergrund dieser Einschränkungen wird in der vorliegenden Arbeit ein Steuerungskonzept entwickelt und evaluiert, das es ermöglicht, den Bearbeitungsprozess über die Manipulation der NC-Programmvorgaben im laufenden Bearbeitungsprozess zu regeln. Dazu interagieren das Programmiersystem, die NC-Steuerung sowie die zu integrierenden Überwachungssysteme. Ziel des Gesamtsystems ist es, die aktuelle Bearbeitungsaufgabe mit ihren prozessbestimmenden Programmvorgaben darzustellen, situationsgerecht Programmalternativen anzubieten, Eingriffe durch den Werker oder eine Regelung zuzulassen, Fehleingaben abzufangen und letztlich die Korrekturen schnell und für den Anwender nachvollziehbar im Bearbeitungsprozess umzusetzen. Dieses System unterstützt den Werker zum einen beim Einfahren neuer Programme und zum anderen bei der Reaktion auf Prozessstörungen in der laufenden Bearbeitung. Dazu müssen die Eingriffsmöglichkeiten ausgehend von den Steuerungsparametern und Sollwerten zusätzlich auf die im NC-Programm kodierten Abläufe, Operationen, Werkzeuge und Technologiedaten erweitert werden. Somit lassen sich im Bearbeitungsablauf das nachfolgend einzuwechselnde Werkzeug neu definieren, die Schnittparameter einer Operation optimieren oder die Bahnauflösung adaptieren.

Introduction

Numerically controlled machine tools are well established in modern production. Referring to the number of metal-cutting machine tools build in Germany their share adds up to more than 85% [VDW04]. Compared to manually or mechanically controlled machine tools they offer a multitude of advantages including automation of machining processes, short cycle times, high reproducibility at constant quality and the flexible manufacturing of different work pieces [WECK01c, KIEF05].

The exploitation of these potentials does not only depend on the numerical control (NC) and the machine tool to be controlled. To a great extend the part program that contains the coding of an idealized process and its adaptation to reality determine the technical result and the profitability of a numerically controlled machining process [EVER94]. Therefore industry continuously asks for systems supporting fast coding of reliable, error prone and optimized NC-programs [BITZ96, STEP99, NCG05] and, on the other hand, for control systems that allow to manually or automatically correct a running machining process [VDW02].

Based on these demands computerized methods for coding NC-programs have been developed ever since the first numerical controls have been introduced. Depending on the site of application and the complexity of the machining tasks different planning, coding and simulation systems are being used in industry. For reasons of costs, work environment and user qualification the functionality of these systems constantly decreases starting with the computer aided systems in the planning department down to the shop floor with their editor based NC-coding. Hence the operator is not adequately supported in coding, testing and correcting NC-programs at the machine tool [NCG93, ENGS02]. As a consequence, not all potentials of process, machine tool and cutting tool are utilized. Moreover, unrecognized coding errors can lead to loss of quality, collisions and damages.

The job of testing a NC-program on a machine tool and of adapting it to the real process is generally executed by the operator at the machine tool and not by specially trained tool setters [LAYG05]. Thereby the operator is facing great demands. He needs to monitor the automated, fast, complex and difficult to observe machining process in order to detect deviations immediately and to start the right countermeasures in time [SCHU00].

Instead of working with comprehensive task and feature descriptions the operator has to think and work with machine oriented, abstract NC commands [SCHI95, BROU00]. These commands are also used to display the machining progress. High level descriptions such as machining operations and strategies are missing as well as a graphical visualization of the manufacturing geometry or the toolpaths. During the machining process the numerical control locks the content of the NC-program. The NC strictly executes the commands line by line. In order to correct the running process only the overrides for spindle speed and feed or simple NC-variables can be manipulated by the operator, Figure 1.1.

In order to change the sequence of working steps, the strategy, the cutting tools or the toolpath the operator has to stop the machining process. Currently this is the only way to correct wrong or sub-optimal NC-programs next to the machine tool or in the planning department. Then the NC-program has to be re-loaded into the NC before it can be started again. Except some variables it is not possible to manipulate the content of the NC-program in parallel to its execution. While the machine tool stops and the NC-program is corrected precious production time gets lost. Moreover it cannot be guaranteed that the correction does not contain additional faults and that the new NC-program is sufficiently optimized.

On the background of these limitations this thesis develops and evaluates a concept that permits the correction of an ongoing machining process by manipulating the NC-program's set-values at runtime. Therefore the NC-programming system, the numerical control as well as the monitoring and control systems interact with each other. The objective of the overall sys-

tem is to describe the manufacturing task with all information needed to understand, execute and manipulate the process. It offers adequate alternatives and enables the operator to adapt the NC-program. Changes are verified, wrong input is rejected and valid corrections are immediately executed and the effect is displayed in a comprehensible way to the operator. This system supports the operator while qualifying a new process or reacting on a process disturbance. Therefore the interaction methods need to be extended from the existing control parameters and set-values to additional data given in the new NC-program, like working steps, operations, tools and technology parameters. Thus in parallel to the machining process the next tool can be redefined, the cutting parameters of an operation can be optimized or the toolpath can be adapted.

2. Stand der Technik

State of the art

Die Grundlage der numerisch gesteuerten Fertigung ist das rechnerbasierte Abarbeiten vordefinierter Steuerprogramme, mit dem Ziel, die Bewegungsachsen, Werkzeugspindeln, Schalt- und Leistungsfunktionen einer Werkzeugmaschine automatisiert anzusteuern [KIEF05]. Das setzt voraus, dass der Bearbeitungsablauf vor seiner Ausführung vollständig geplant und als Abfolge zeitlich aufeinander abgestimmter, steuerungskonformer Weg-, Schalt- und Technologieanweisungen kodiert worden ist. Da die einzelnen Programmanweisungen teilweise voneinander abhängen und sich gegenseitig beeinflussen, muss die Steuerung das Programm im Vorlauf zur eigentlichen Bearbeitung einlesen, befehlsübergreifend auswerten und in synchrone Antriebs- und Schaltsignale umsetzen. Dazu werden die im Steuerprogramm kodierten Anweisungen in den Datenspeicher der Steuerung geladen und während der Verarbeitung für externe Zugriffe gesperrt. Folglich muss bei Prozessstörungen oder Programmfehlern der Anwender die Bearbeitung abbrechen, um das Programm aus der Steuerung zu entladen und Korrekturen am programmierten Ablauf vornehmen zu können. Parallel zur Programmausführung kann der Werker den Prozess nicht über die Programmvorgaben, sondern nur über Regelfunktionen der Steuerung, wie beispielsweise die Overrides für den Bahnvorschub oder die Spindeldrehzahl beeinflussen. Da die Werkzeugbewegungen im Vorfeld der Bearbeitung berechnet und als einfache Verfahranweisungen kodiert werden, kann die Steuerung die Bahnen zur Laufzeit nicht an eine geänderte Strategie oder ein anderes Werkzeug anpassen.

Bevor die Aufgabenstellung detailliert und ein System zur Lösung dieser Einschränkung entwickelt werden kann, wird nachfolgend der aktuelle Stand der Technik in den Bereichen der NC-Programmierung, Programmverarbeitung und Prozessregelung analysiert. Dazu wird zunächst herausgearbeitet, welche Informationen über das NC-Programm an der Werkzeugmaschine und in der NC-Steuerung bereitgestellt werden. Daran schließt sich eine Analyse der Programmverarbeitung und des Aufbaus von aktuellen NC-Steuerungen an. Abschließend werden bestehende Lösungen zur Prozessüberwachung und -regelung dargestellt.

2.1. NC-Programmierung

NC-programming

Die NC-Programmierung umfasst alle Arbeitsschritte, die notwendig sind, um ein Werkstück auf einer numerisch gesteuerten Maschine automatisiert fertigen zu können. Hierzu gehören die Ermittlung der geometrischen, technologischen und ablauforientierten Bearbeitungsinformationen sowie deren steuerungskonforme Kodierung in einem NC-Programm. In Abhängigkeit von der verfügbaren Systemunterstützung wird das NC-Programm vor der Ausführung simuliert, um die Taktzeiten zu bestimmen und das Programm auf Fehler zu überprüfen. An der Maschine wird dann das NC-Programm schrittweise im Luftschnitt oder mit reduzierten Verfahrgeschwindigkeiten erprobt und optimiert.

Die geometrischen und operativen Planungsinformationen, die während der NC-Programmierung als Eingangsdaten vorliegen und zusätzlich erzeugt werden, werden nur teilweise in das NC-Programm übertragen. In der Regel enthält ein NC-Programm nur noch eine Sequenz elementarer Weg-, Schalt- und Technologiebefehle, Bild 2.1. Dieses Format ist an den Funktionsumfang der NC-Steuerung angepasst und für eine schnelle Programmumsetzung optimiert. Die fehlenden Geometrie- und Planungsinformationen schränken jedoch die Möglichkeit einer aufgaben- und prozessorientierten Programmänderung, Bearbeitungsverfolgung, Überwachung und Regelung an der Maschine und im Prozess ein [BROU00, WECK03a].

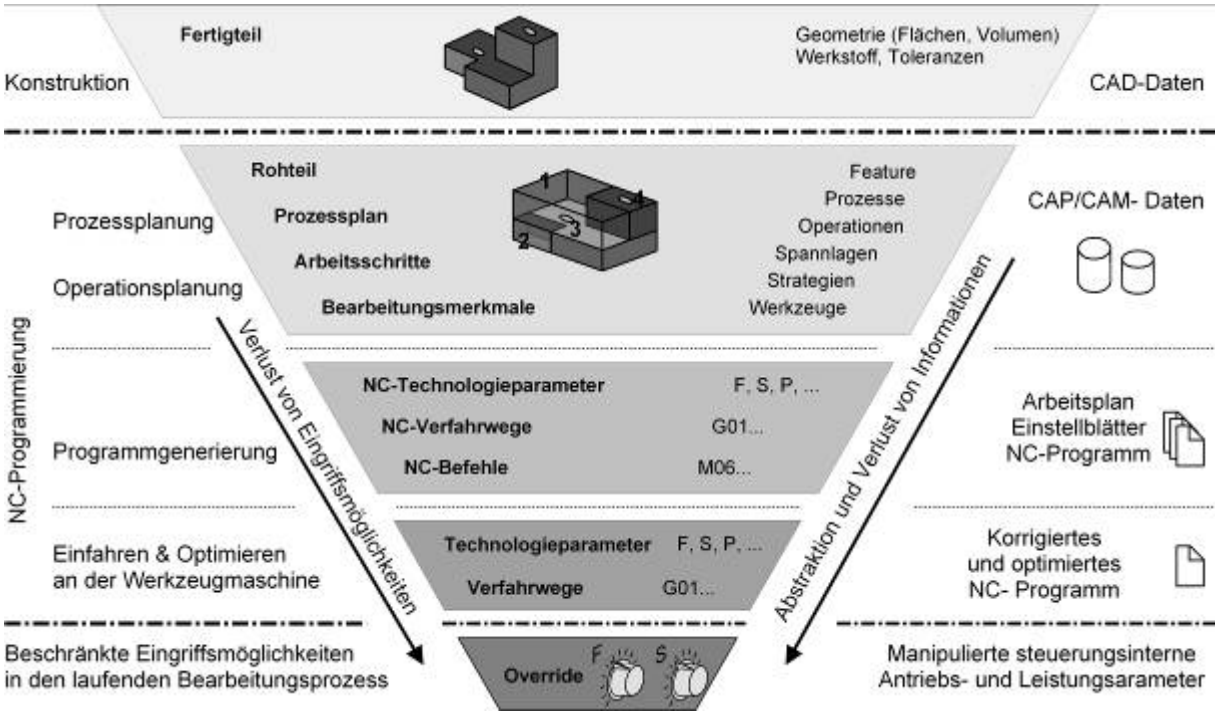


Bild 2.1: Informationsverlust im Rahmen der NC-Programmierung
Information loss during NC-programming

Nach Eversheim lässt sich die NC-Programmierung in die Prozessplanung, Operationsplanung und die NC-Programmgenerierung untergliedern [EVER94]. Obgleich alle diese Schritte im Vorfeld der Bearbeitung und oftmals fern der numerisch gesteuerten Maschine in der Arbeitsvorbereitung ablaufen, sind sie auch für eine Prozessüberwachung und -regelung zur Programmlaufzeit wichtig, da sie den steuerungstechnischen Bearbeitungsablauf vorgeben. Nachfolgend werden daher die verschiedenen Verfahren zur NC-Programmierung kurz dargestellt. Im Anschluss wird aufgezeigt, welche Informationen in den einzelnen Planungsschritten vorliegen, mit welchem Verlust diese auf elementare Steuerungsbefehle reduziert werden und inwiefern dadurch die Möglichkeiten zu Programmänderungen an der Maschine und im Prozess eingeschränkt werden.

2.1.1. Verfahren zur NC-Programmierung

Methods for NC-programming

Die Erstellung der Steuerinformationen ist seit der Entwicklung der ersten numerisch gesteuerten Werkzeugmaschine eine Schwachstelle der NC-Technik [HENN97]. Ausgehend von Zeichnungen, Maschinen- und Werkzeugdaten sowie dem Fertigungswissen muss die Bearbeitungsaufgabe in Form von elementaren Steuerungsbefehlen beschrieben werden. Neben der direkten, manuellen Programmierung wurden daher bereits früh rechnerbasierte Programmierverfahren entwickelt, um den Programmierer durch Automatismen bei der problemorientierten Beschreibung der Bearbeitungsaufgabe, der Berechnung der Technologieparameter und Verfahrwege sowie dem finalen Kodieren der Weg-, Schalt- und Technologiebefehle zu unterstützen. Der Funktionsumfang dieser Systeme wurde fortlaufend weiterentwickelt und hat sich ausgehend von der Arbeitsvorbereitung bis in die Werkstatt und die Benutzerschnittstelle der Steuerung etabliert, Bild 2.2. Heute bieten praktisch alle Steuerungshersteller Editoren und teilweise grafisch unterstützende, dialoggeführte Programmierhilfen in den Benutzungsoberflächen ihrer NC-Steuerungen an.

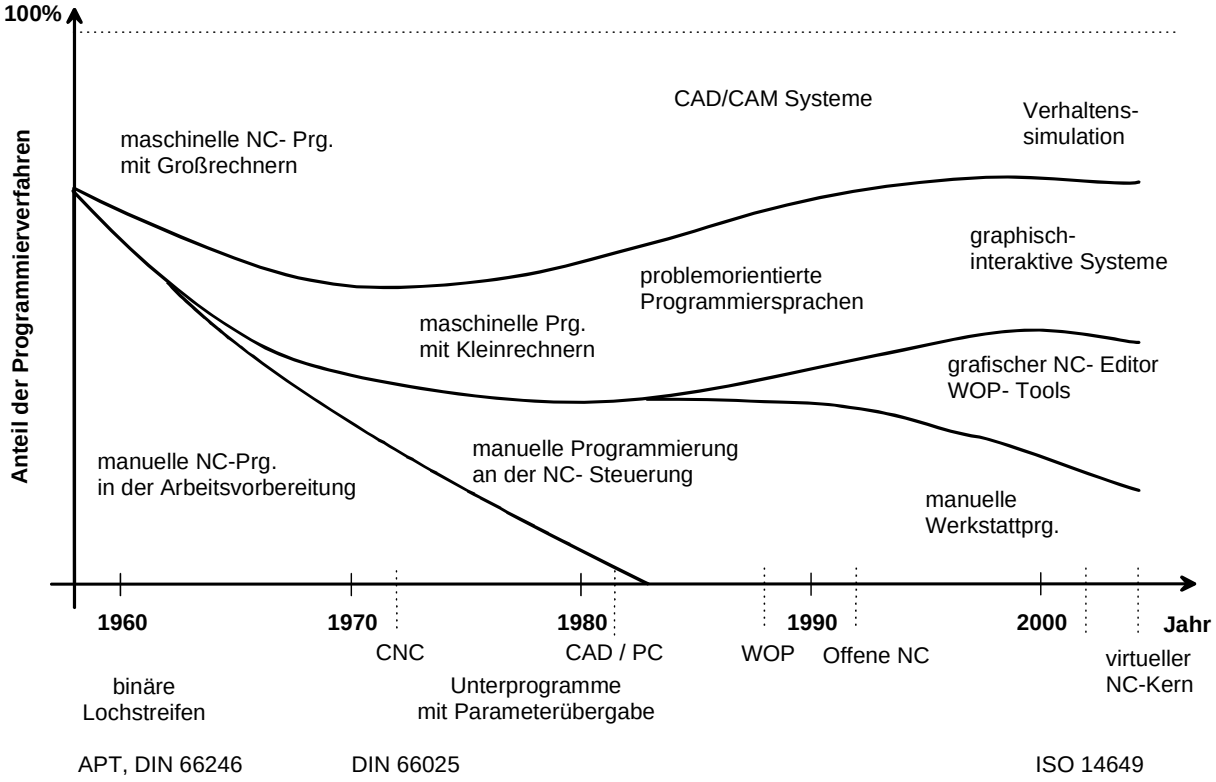


Bild 2.2: Systemunterstützung bei der NC-Programmierung (angelehnt an [BENK95])
System support in NC-programming

Während einfache Geometrien und Bearbeitungsaufgaben auch heute noch direkt mit Hilfe eines Texteditors in Form von NC-spezifischen Befehlen programmiert werden, erfordern umfangreichere Programmieraufgaben, wie die Freiformflächenbearbeitung, rechnerbasierte Hilfen zur Geometriedatenverarbeitung, Ablaufplanung, Technologieauswahl und Schnittaufteilung. Je nach Komplexität dieser Hilfssysteme und in Abhängigkeit von der Qualifikation der Planer und Programmierer werden sie in der Arbeitsvorbereitung, der Werkstatt oder integriert in der Benutzeroberfläche der NC-Steuerung eingesetzt. Die einzelnen Verfahren werden nachfolgend nur kurz vorgestellt. Eine detaillierte Beschreibung findet sich u.a. in [WECK01d] und [EVER02].

2.1.1.1. Manuelle NC-Programmierung

Manual NC-programming

Ausgehend von den spezifischen Programmcodes, d.h. dem NC-Programmierschlüssel einer Steuerung und Werkzeugmaschine, werden bei der manuellen Programmierung die NC-Codewörter und Parameter von Hand berechnet und an einem Texteditor in Form eines NC-Programms eingeben, Bild 2.3 rechts. Insbesondere bei einfachen Aufgaben ohne komplizierte Schnittaufteilungen oder Achsbewegungen lassen sich so Verfahrwege und Anweisungen schnell kodieren. Wenn der Programmierer gleichzeitig der Maschinenbediener ist, kann er das Programm direkt an der Maschine erstellen, testen und optimieren.

Zudem wird die manuelle Programmierung bei der Korrektur und zum Optimieren von bestehenden Programmen eingesetzt. Dabei werden besondere Anforderungen an den Facharbeiter gestellt. So muss er unter anderem das NC-Programm rein textbasiert interpretieren, um die Bearbeitungsaufgabe und deren technologischen Ablauf rekonstruieren zu können [SCHI95]. Wie in Bild 2.1 dargestellt ist, hat er über das NC-Programm keinen Zugriff auf die für ihn verständlichen, aufgaben- und operationsorientierten Konstruktions- und Planungsdaten.

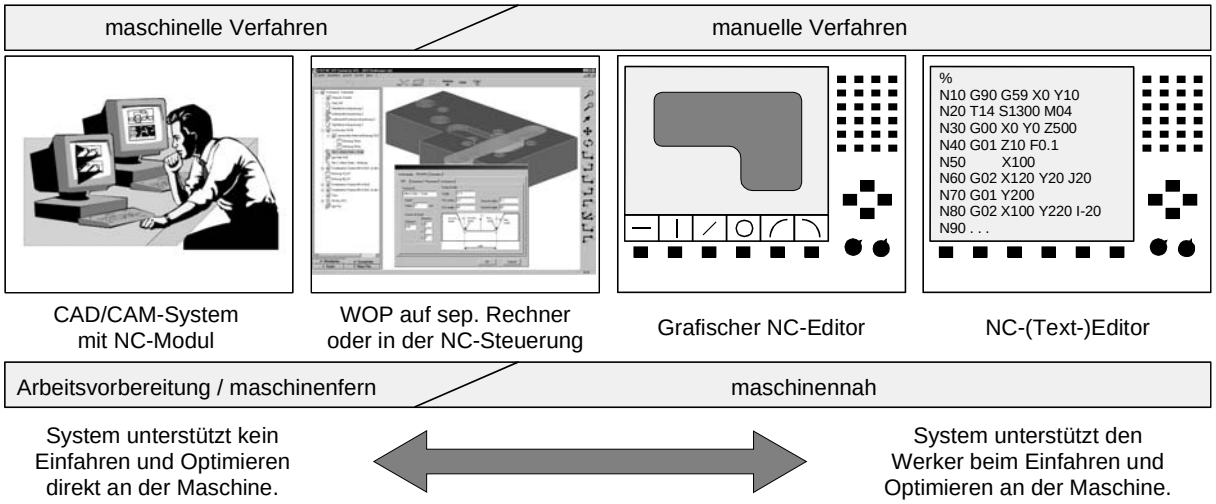

maschinelle Verfahren
manuelle Verfahren
%
N10 G90 G59 X0 Y10
N20 T14 S1300 M04
N30 G00 X0 Y0 Z500
N40 G01 Z10 F0.1
N50 X100
N60 G02 X120 Y20 J20
N70 G01 Y200
N80 G02 X100 Y220 I-20
N90 . . .
CAD/CAM-System mit NC-Modul
WOP auf sep. Rechner oder in der NC-Steuerung
Grafischer NC-Editor
NC-(Text-)Editor
Arbeitsvorbereitung / maschinenfern
maschinennah
System unterstützt kein Einfahren und Optimieren direkt an der Maschine.
System unterstützt den Werker beim Einfahren und Optimieren an der Maschine.

Bild 2.3: NC-Programmierverfahren
Methods for NC-programming

Bei umfangreichen und komplexen Aufgaben stößt die manuelle Programmierung schnell an ihre Grenzen. Aufgrund der fehlenden Systemunterstützung bei der Geometrieverarbeitung, der Arbeitsfolgebestimmung, der Technologie- und Strategieauswahl sowie der Generierung der elementaren Steuerungsanweisungen ist das manuelle Verfahren zeitintensiv und fehleranfällig. Der Programmierer verliert in langen Programmen schnell den Überblick, da ihm Hilfen, wie beispielsweise strukturierte Geometrie-, Operations- und Werkzeugbeschreibungen fehlen. Die Folge sind Programmierfehler, die während der Bearbeitung oftmals zu schwerwiegenden Folgefehlern am Werkstück, Werkzeug oder an der Maschine führen [WECK01d].

Aus der Sicht eines Softwarearchitekten hat die manuelle Programmierung den Nachteil, dass sie keinem definierten Muster folgt. Jeder NC-Programmierer hat seine individuelle Art die Bearbeitung durch NC-Codes zu beschreiben. Zudem fehlen Rohteilinformationen, so dass die automatisierte Rekonstruktion der Bearbeitungsgeometrie und -aufgabe nur eingeschränkt möglich ist. Die am Markt befindlichen Systeme zur Überwachung und Optimierung beschränken sich daher auf die Auswertung der programmierten Verfahrwegbefehle und Vorschübe [MAV05, SCHI05]. Kollisionen werden in der Regel auf der Basis zusätzlicher, steuerungsspezifischer Kollisionsräume und Werkzeugdefinitionen überwacht. Die Steuerung stoppt in diesem Fall die Achsen, bevor sie in einen gesperrten Bereich einfahren [NCFE05]. Sie kann jedoch die Bearbeitungsbahn nicht zur Kollisionsvermeidung umplanen. Andere Systeme lesen zusätzlich CAD-Datensätze ein, um die Werkzeugbewegungen mit der zu fertigenden Kontur abgleichen und offline optimieren zu können [CGTE04]. Am Markt sind jedoch keine Systeme verfügbar, mit denen sich ausgehend von einem NC-Programm die Operationsfolge, die Strategie, die Technologie und die finale Schnittaufteilung während der Bearbeitung korrigieren und optimieren ließen.

2.1.1.2. Maschinelle NC-Programmierung in der Arbeitsvorbereitung

Computer aided programming in the process planning

Die maschinelle, d.h. die rechnerunterstützte Programmierung orientiert sich an der zu bearbeitenden Werkstückgeometrie, Bild 2.3 links. In einem ersten Schritt werden die Roh- und Fertigteilgeometrie in den Systemen grafisch-interaktiv erstellt, aus CAD-Geometriedatensätzen importiert oder, bei CAD/CAM-Systemen, direkt aus dem CAD-System übernommen. Daran schließt sich die Definition der zu bearbeitenden Bereiche, d.h. der zu fertigenden Flächen und Merkmale an. Hierbei wird der NC-Programmierer zunehmend durch fertigungsgerechte Konstruktionsmerkmale aus dem CAD [HOFF05], eine automatische Featureerken-

nung oder die Generierung von Fertigungshilfsmerkmalen durch das CAP/CAM-System unterstützt [KOCH05]. In einem nächsten Schritt werden ausgehend von den Bearbeitungsmerkmalen, die Abarbeitungsreihenfolge und die einzelnen Fertigungsoperationen mit den dazugehörigen, Strategien, Werkzeugen und Technologieparametern geplant. Bei diesen planerischen Aufgaben unterstützt das Programmiersystem den Anwender durch Vorschläge und Auswahloptionen basierend auf Werkzeug- und Technologiedatenbanken.

Bis zu diesem Punkt der NC-Programmierung speichern die Systeme die Planungsergebnisse in ihren proprietären, internen Datenformaten. Die Bearbeitung wird umfassend und zusammenhängend durch die Roh- und Fertigteilgeometrie sowie die Operationsdaten beschrieben. Die Datenstruktur entspricht dem arbeitsschritt- und merkmalorientierten Denken des Anwenders. Dabei wird die CAD-Geometrie fertigungsgerecht umgesetzt und irreversibel in das Format des Planungssystems konvertiert. Änderungen an der Geometrie müssen manuell im Konstruktionsmodell oder in der Schnittaufteilung des NC-Programms nachgehalten werden [HENN97]. Eine Alternative bieten die sich in der Praxis durchsetzenden, gekoppelten CAD/CAM Systeme. Ausgehend von einem gemeinsamen Datenmodell werden Änderungen an der Geometrie direkt in die Konstruktion und in die Bearbeitung übernommen [LAUS01].

Als neutrales Ausgabeformat generieren die Planungs- und Programmiersysteme problemorientierte, auf die spätere Ausführung ausgerichtete Aufgabenbeschreibungen, beispielsweise nach APT (Automatically Programmed Tools, DIN66246). In diesen Datensätzen sind nur noch vereinfachte Rohteil-, Ablauf- und Operationsinformationen enthalten. Ein weiteres Zwischenformat ist die Cutter Location Data nach DIN 66215, in der die Bearbeitungsaufgabe auf einzelne Werkzeugbewegungen und Schaltanweisungen heruntergebrochen und maschinenneutral kodiert wird.

Im letzten Schritt löst ein Postprozessor die Bearbeitungsmerkmale, Operationsvorgaben und Anweisungen steuerungsgerecht auf. Das Ergebnis ist eine maschinen- und steuerungsspezifische Sequenz von Verfahrwegen, Technologie- und Schaltanweisungen (Bild 2.4).

Die eingesetzten Postprozessoren sind keine reinen Umsetzer zwischen zwei Formaten. Ihre Aufgabe besteht darin, die Ablauf-, Operations- und Geometriedaten der Planung zu interpretieren und technologisch optimierte, maschinen- und steuerungsspezifische Weg-, Technologie- und Schaltanweisungen zu berechnen und auszugeben. Typische Beispiele sind das Auflösen von Taschenproblemen, Bohroperationen oder das Zusammenfassen und Optimieren von Verfahrbewegungen. Folglich ist in Postprozessoren ein großes Maß an technologiespezifischem Know-how implementiert. Es gibt Postprozessoren mit eigener Benutzeroberfläche, offenen Schnittstellen zur Anpassung und Erweiterung der Umsetzungslogik und einer grafischen Anzeige zur Simulation und Verifikation der generierten Programme [FRAN02]. Sie sind entweder in die CAM-Systeme integriert, arbeiten als eigenständige Applikationen oder werden in der NC-Steuerung in Form von Compilern vor der Programminterpretation aufgerufen [INDR96, HOFF05].

Das Ergebnis der maschinellen NC-Programmierung ist eine Abfolge von Weg-, Schalt- und Technologiefunktionen. Die hochwertigen Informationen der Teilsysteme werden nicht in das NC-Programm übertragen und stehen dem Werker an der Maschine aufgrund einer fehlenden Integration der CAD/CAM Module in die Benutzeroberfläche der NC nicht zur Verfügung. Programmänderungen an der Maschine oder Manipulationen im Prozess sind auf die NC-Befehle begrenzt.

Aus der Sicht der Anwender sind Postprozessoren teure, wartungs- und pflegeintensive Systeme. Die Anwender fordern daher eine Vereinheitlichung der Funktionen und versprechen sich davon eine vereinfachte, kostengünstige Lösung, in der Planungs- und NC-Programmdaten beliebig zwischen CAM und NC-Steuerung austauschbar sind und in der Änderungen durchgängig vom CAD bis zur NC nachgehalten werden [STEP99, WEYR00, DEIT04].

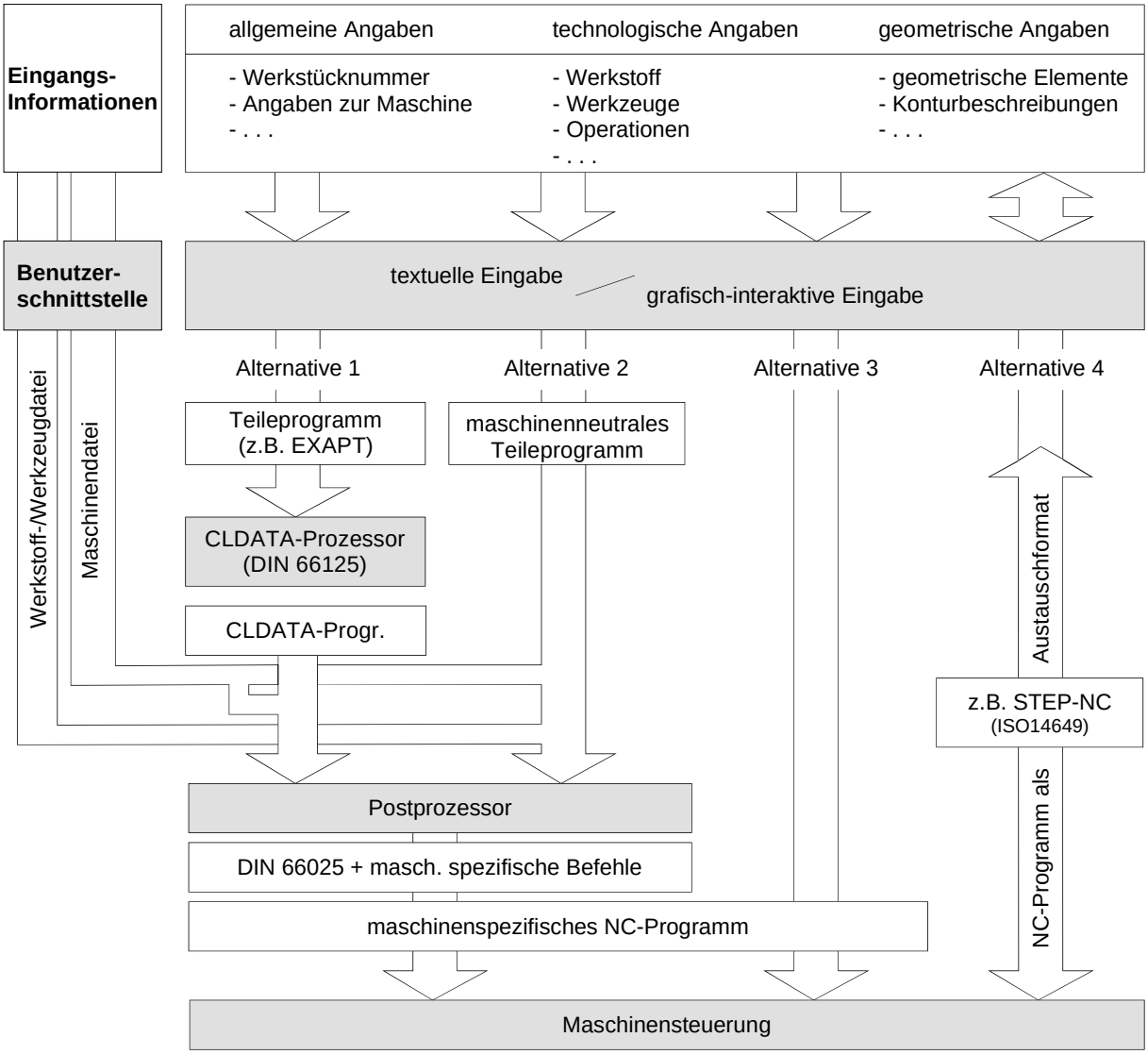


Bild 2.4: Alternative Wege der maschinellen NC-Programmgenerierung, nach [WECK01d]
Alternative methods of computer aided NC-programming

Werkstattorientierte NC-Programmierung

In der Werkstatt und an der Maschine werden aus Gründen des Arbeitsumfelds und der hohen Softwarekosten praktisch keine CAD/CAM Systeme eingesetzt. Dennoch besteht der Bedarf, den Werker bei der Überprüfung und Korrektur bestehender Programme sowie dem Erstellen neuer Bearbeitungsprogramme vor Ort zu unterstützen. Hierzu wurden werkstattorientierte Programmiersysteme (WOP) entwickelt. Sie unterstützen den Programmierer auf einem externen Rechnerarbeitsplatz oder direkt in der Benutzeroberfläche der NC-Steuerung. Je nach Funktionsumfang der WOP Systeme lassen sich CAD Daten importieren, Bearbeitungsgeometrien dialogbasiert erstellen und Operationen grafisch interaktiv planen. Das System generiert anschließend steuerungsspezifische Befehle und Zyklen. Deren Korrektur, Ergänzung und Optimierung setzt voraus, dass der Anwender die manuelle Programmierung beherrscht [DOMB93]. Die in die Benutzeroberfläche der Steuerung integrierten WOP Systeme bieten zudem Bedienfunktionen, mit denen sich das NC-Programm starten, schrittweise ausführen und der Abarbeitungsfortschritt grafisch verfolgen lässt [NCG93].

Die Kopplung der WOP Systeme ist jedoch auf die Programmierschnittstelle der Steuerung sowie die genannten Steuerungsfunktionen begrenzt. Ein Zugriff auf das NC-Programm und die darin kodierten Strategie- und Technologievorgaben ist zur Laufzeit nicht möglich.

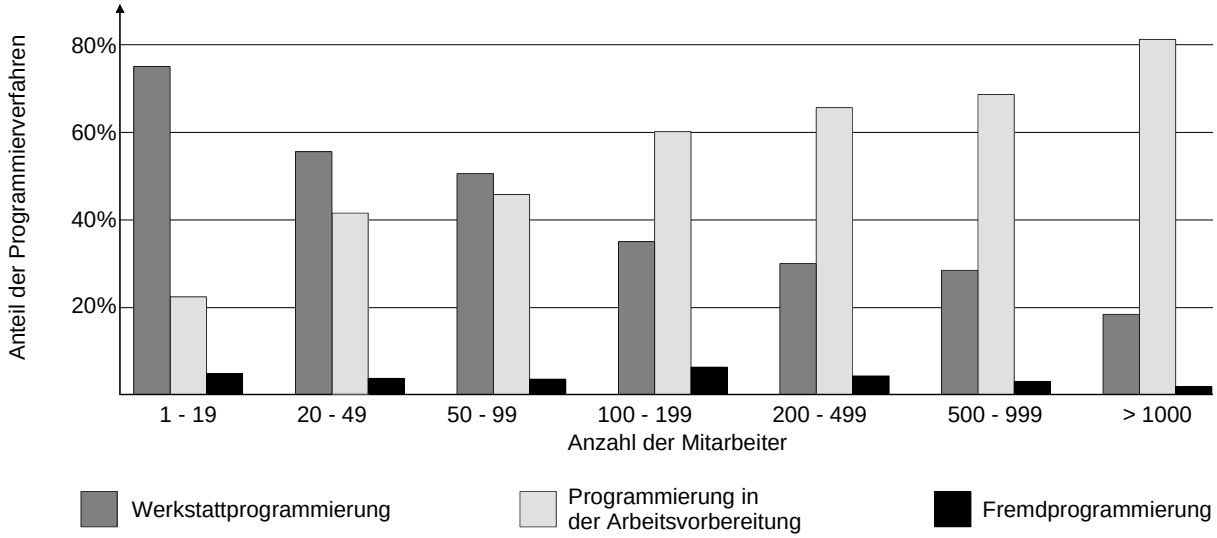


Bild 2.5: Eingesetzte NC-Programmierverfahren in Abhängigkeit von der Unternehmensgröße, nach [NCG93, LAY05]

Use of NC-programming methods depending on industry's size

Studien der NC-Gesellschaft und der Fraunhofer Gesellschaft spiegeln die fortlaufende Bedeutung der manuellen und WOP-unterstützten NC-Programmierverfahren wieder [NCG93, LAYG05]. Demnach erfolgt in Kleinbetrieben über 70% der NC-Programmierung in der Werkstatt (Bild 2.5). Und selbst in größeren Firmen, die ihre Programmieraufgaben primär in der Arbeitsvorbereitung abwickeln, hat die Werkstattprogrammierung mit annähernden 20% einen großen Anteil. Dieser hohe Anteil ist darauf zurückzuführen, dass ein großer Bedarf an maschinenahen Programmkorrekturen und -optimierungen besteht [BEUK99].

2.1.2. NC-Programmierschnittstelle

NC-programming interface

Am Ende aller NC-Programmierverfahren steht das Generieren der Steuerinformationen in Form eines NC-Programms. Die zur Programmgenerierung verfügbaren NC-Befehle, Parameter und das Programmformat hängen von dem Funktionsumfang der anzusteuernden Maschine sowie der NC-Steuerung ab, die das NC-Programm interpretiert und verarbeitet. Der zugelassene Befehls- oder Sprachumfang bildet die NC-Programmierschnittstelle einer Steuerung. Sie wird in der Programmieranleitung der jeweiligen Maschine dokumentiert.

Aufgrund der Vielzahl der maschinen-, steuerungs- und anwenderspezifischen Anforderungen gibt es im industriellen Einsatz keine einheitliche, allgemeingültige NC-Programmierschnittstelle. Vielmehr gibt es einen Standard, der von den Steuerungsherstellern als Vorlage genutzt wird und von dem sie ihre spezifischen Schnittstellen ableiten. International hat sich hierzu die DIN 66025 / ISO 6983 durchgesetzt [DIN66025].

Darüber hinaus ermöglichen viele Hersteller die Programmierschnittstelle ihrer Steuerung zu parametrieren und zu erweitern. Diese Offenheit nutzen die Maschinenhersteller bei der Zuordnung der Maschinenachsen und -funktionen zu den Programmcodes und, um eigene Routinen in der NC-Steuerung implementieren und aufrufen zu können.

2.1.2.1. DIN 66025 – sequentielle, befehlsorientierte NC-Programmierung

DIN 66025 / ISO 6938 – sequential, command oriented NC-programming

Weltweit hat sich die DIN 66025 (international: RS 274, ISO 6938) als Grundlage für die Programmierung numerisch gesteuerter Werkzeugmaschinen etabliert. Sie definiert allgemeingültige Grundbefehle und gibt eine Syntax sowie Richtlinien zur Erweiterung ihres Befehlsumfangs vor. Ausgehend von der DIN 66025 ist es daher möglich, eigene, spezialisierte Programmierschnittstellen abzuleiten und diese an die individuellen Technologie- und Funktionsanforderungen der Steuerung und der Maschine anzupassen.

Ein NC-Programm nach DIN 66025 besteht aus dem Programmanfang, einer beliebigen Anzahl von NC-Sätzen, die den Arbeitsablauf schrittweise beschreiben, und dem Programmende. Jeder Satz kodiert ein Bewegungssegment und / oder eine Maschinenoperation. Er besteht aus einem oder mehreren Worten, die sich aus Adressbuchstaben und Ziffernfolgen zusammensetzen. Die Adressbuchstaben unterscheiden sich nach geometrischen Anweisungen (z.B. A, B, C, X, Y, Z, U, V, W, usw.), technologischen Anweisungen (F, S, T), Fahranweisungen (G), Schaltbefehlen (M), Korrekturaufrufen (H) und nicht normierten, d.h. dem Steuerungs- und Maschinenhersteller überlassenen Funktions-, Zyklen- und Unterprogrammaufrufen.

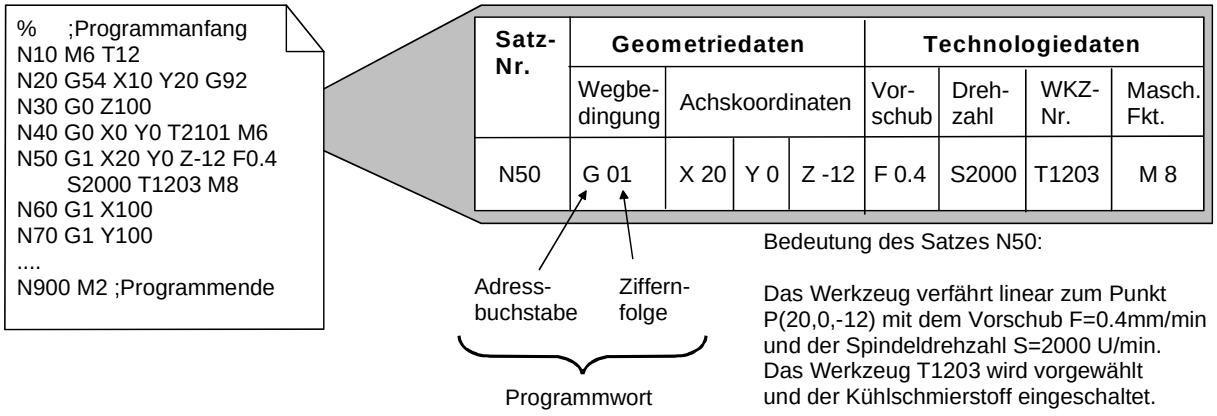


Bild 2.6: Aufbau einer Verfahrweganweisung nach DIN 66025
Movement command according to DIN 66025 / ISO 6938

Der einfache Befehlsumfang der DIN 66025 ist auf die „am meisten verwendeten Wegbedingungen und Zusatzfunktionen“ begrenzt [DIN66025]. Höherwertige Beschreibungen des Fertigungsmerkmals, der Arbeitsschritte oder der Bearbeitungsoperation fehlen. Diese sind nur noch implizit durch die programmierten Verfahrwege und die Technologieparameter enthalten. Die Steuerungshersteller haben daher die Grundbefehle des Standards an ihre Bedürfnisse angepasst (z.B. herstellerspezifische Nullpunktverschiebungen) und um eigene Befehle ergänzt (z.B. Spline-Funktionen, Hochsprachenelemente und Bearbeitungszyklen).

2.1.2.2. NC-Zyklen

NC-cylces

Mit den herstellerspezifischen Bearbeitungszyklen, auch NC-Zyklen genannt, lassen sich verfahrenstypische Bearbeitungsmuster mithilfe weniger Parameter programmieren. Die Auflösung der NC-Zyklen in Schalt-, Technologie- und Wegbefehle übernimmt ein der Steuerung vorgeschalteter Compiler oder direkt die Programmauflösung der NC-Steuerung. Viele Steuerungshersteller bieten umfangreiche Zyklenbibliotheken an, so dass der Anwender die Bearbeitungsaufgabe vollständig zyklenbasiert programmieren kann, ohne DIN Befehle kennen zu müssen. Dabei unterstützen sie den Anwender durch grafische Darstellungen und dialoggeführte Parameterabfragen [HEID04, SIEM02]. Neben einfachen Konturzyklen gibt es merk-

mals- und operationsorientierte Zyklen. Mit ihnen lassen sich typische Merkmalsgeometrien und Operationen, wie Tieflochbohrungen und Taschenbearbeitungen, mithilfe weniger Parameter beschreiben und im NC-Programm kodieren. Ausgehend von diesen Parametern und dem dahinterliegenden Modell lassen sich über Zyklen höherwertige Planungs- und Operationsdaten in die NC-Steuerung einlesen und verarbeiten. Da die NC-Zyklen nicht standardisiert sind, können sie nicht zwischen verschiedenen NC-Steuerungen ausgetauscht werden. Oftmals sind sie zudem anwenderspezifisch in die Steuerung integriert worden, so dass der Zyklus nur auf dieser Steuerung und ihrem Softwarestand läuft. Aufgrund dieser Einschränkungen unterstützen steuerungsneutrale Programmiersysteme NC-Zyklen nur selten.

2.1.2.3. Defizite der aktuellen DIN Programmierung

Deficits of today's ISO 6938 NC-programming

Die DIN Programmierung und ihre Erweiterungen haben zusammenfassend folgende grundlegende Nachteile:

- Geringes semantisches Niveau, das sich an den Steuerungsfunktionen, nicht aber am Verständnis des Anwenders oder am Prozess orientiert.
- Änderungen müssen manuell in der Planung nachgehalten werden, da Konstruktions- und Planungsdaten nicht im NC-Programm enthalten oder mit diesem verknüpft sind.
- Aufgrund der nur im Grundumfang standardisierten NC-Codes sind die Steuerungsfunktionen nur bedingt vergleichbar. Die Befehlsausführung ist somit steuerungsspezifisch.
- Es ist kein Programmaustausch zwischen verschiedenen Maschinen, Steuerungen und Programmiersystemen möglich. Die Systembindung schränkt die Flexibilität ein.
- Es fehlen standardisierte Elemente und Zusatzinformationen, die als Grundlage einer Prozessverfolgung, -überwachung und -regelung ausgewertet werden könnten.

2.1.3. Einfahren und Korrektur von NC-Programmen

Optimization and correction of NC-programs

Sowohl bei manuellen als auch bei maschinellen Programmierverfahren können Programmierfehler nicht ausgeschlossen werden. Zur Absicherung wird das erstellte NC-Programm daher simuliert und an der Maschine schrittweise erprobt. Dazu werden unterschiedliche Verfahren angewendet:

- Softwarebasierte Ablaufsimulation und Kollisionsüberprüfung.
- Leerfahrt ohne Werkzeug und Werkstück, so dass keine Kollisionen auftreten können.
- Luftschnitt ohne aufgespanntes Werkstück, um das Kollisionsrisiko zu minimieren.
- Programmausführung im Einzelsatzmodus, so dass der Einrichter vor jedem auszuführenden NC-Satz die Bearbeitung quittieren, d.h. die Bearbeitung freigeben muss.
- Programmausführung mit reduzierten Achsvorschüben.
- Programmausführung mit weichen Versuchswerkstoffen (z.B. Ureol).

Während des Einfahrens eines NC-Programms bestimmen die Übersichtlichkeit des Programmcodes, der Inhalt seiner Anweisungen sowie die an der Maschine verfügbaren Programmierhilfen, wie gut sich das Programm durch den Werker nachvollziehen und wie schnell es sich vor Ort korrigieren lässt. Elementare Befehle und die fehlenden Informationen zur Bearbeitungsaufgabe, -geometrie und Operation erschweren das Auffinden von Fehlern und Optimieren des NC-Programms.

Parallel zur Programmausführung sind die Eingriffsmöglichkeiten in den Prozess auf Steuerungsfunktionen, wie die Overrides begrenzt. Darüber lassen sich beispielsweise die Bahngeschwindigkeit, die Spindeldrehzahl, die Laserleistung oder der Strom einer EDM Bearbeitung beeinflussen. Das NC-Programm und seine Abarbeitung sind dagegen für Zugriffe durch den Facharbeiter gesperrt. Lediglich über die offenen Softwareschnittstellen einzelner Steuerungshersteller ist eine Manipulation der noch nicht umgesetzten NC-Sätze und den darin kodierten Parametern möglich [BOSC04, SIEM03a].

Eingriffe, die sich nicht auf elementare Steuerungsparameter zurückführen lassen, erfordern den Abbruch der Programmausführung, um das NC-Programm korrigieren zu können. Beispiele sind die Abarbeitungsreihenfolge, die Bahnführung oder das einzusetzende Werkzeug. Hierbei geht wertvolle Produktionszeit verloren. Einzelne Steuerungshersteller bieten daher Satz-Vorlauffunktionen an, um das NC-Programm nicht nach jeder Korrektur erneut vom Programmanfang aus starten zu müssen. Zudem gibt es Steuerungen, an denen der Facharbeiter das NC-Programm nach einer Programmunterbrechung direkt manipulieren und fortsetzen kann [SIEM04]. Diese Funktion setzt jedoch voraus, dass die zu manipulierenden Befehle und Zyklen noch nicht vorverarbeitet sind oder aktuell abgearbeitet werden.

2.2. NC-Steuerungen und Programmverarbeitung

Numerical controls and NC-program processing

An einer Werkzeugmaschine kommen zwei Steuerungssysteme zum Einsatz. Die Ablaufsteuerung und die NC-Steuerung. Ein grundlegender Unterschied zwischen den beiden Steuerungen liegt in ihrer Programmierung. Das Programm der Ablaufsteuerung (SPS) wird im Datenspeicher der Steuerung fest kodiert. Es umfasst die Standardfunktionen der Maschine sowie Sicherheitsabfragen, die nach einem immer gleichen Muster abgearbeitet werden, ohne auf den Bearbeitungsprozess zu reagieren. Dagegen verarbeitet die NC-Steuerung kein statisches Ablaufprogramm, sondern ein frei programmierbares, alphanumerisch kodiertes Bearbeitungsprogramm. In diesem Bearbeitungsprogramm sind die Art und die Reihenfolge der zur Herstellung eines Werkstücks notwendigen Fertigungsschritte kodiert [WECK01d]. Folglich sind Kernfunktionen der NC-Steuerung die Interpretation und steuerungstechnische Umsetzung dieser NC-Codes und Parameter.

2.2.1. Ablauf der NC-Programmverarbeitung

Procedure of NC-program processing

Die NC-Programmverarbeitung umfasst alle Schritte, die notwendig sind, um die im NC-Programm kodierten Befehle und Technologieparameter einzulesen, zu interpretieren und in Sollwerte umzusetzen. Dabei werden die Vorgaben des NC-Programms sequentiell eingelesen, schrittweise weiterverarbeitet und als Stellgrößen an die Antriebe oder die SPS weitergeleitet, Bild 2.7. Zuletzt überwacht die NC-Steuerung die Ausführung der Stellgrößen und synchronisiert darüber die nachfolgend auszuführenden Befehle. Adaptive Steuerungen, auf die in Kapitel 2.3 eingegangen wird, führen zudem Prozessgrößen in die Sollwertgenerierung zurück.

Anwahl und Vorverarbeitung des NC-Programms

Die NC-Programmverarbeitung beginnt in der Regel bereits, wenn der Werker die Programmdatei anwählt. Dadurch wird es in den Steuerungsspeicher geladen und für Zugriffe durch Dritte gesperrt. Je nach Steuerung werden gleichzeitig erste Routinen zur Programmüberprüfung und -umsetzung aufgerufen. Typische Beispiele sind syntaktische Überprüfun-

gen der kodierten Befehle oder Compiler, die anwenderspezifische Programmelemente auflösen und in elementare, steuerungskonforme NC-Codes übersetzen [INDR96].

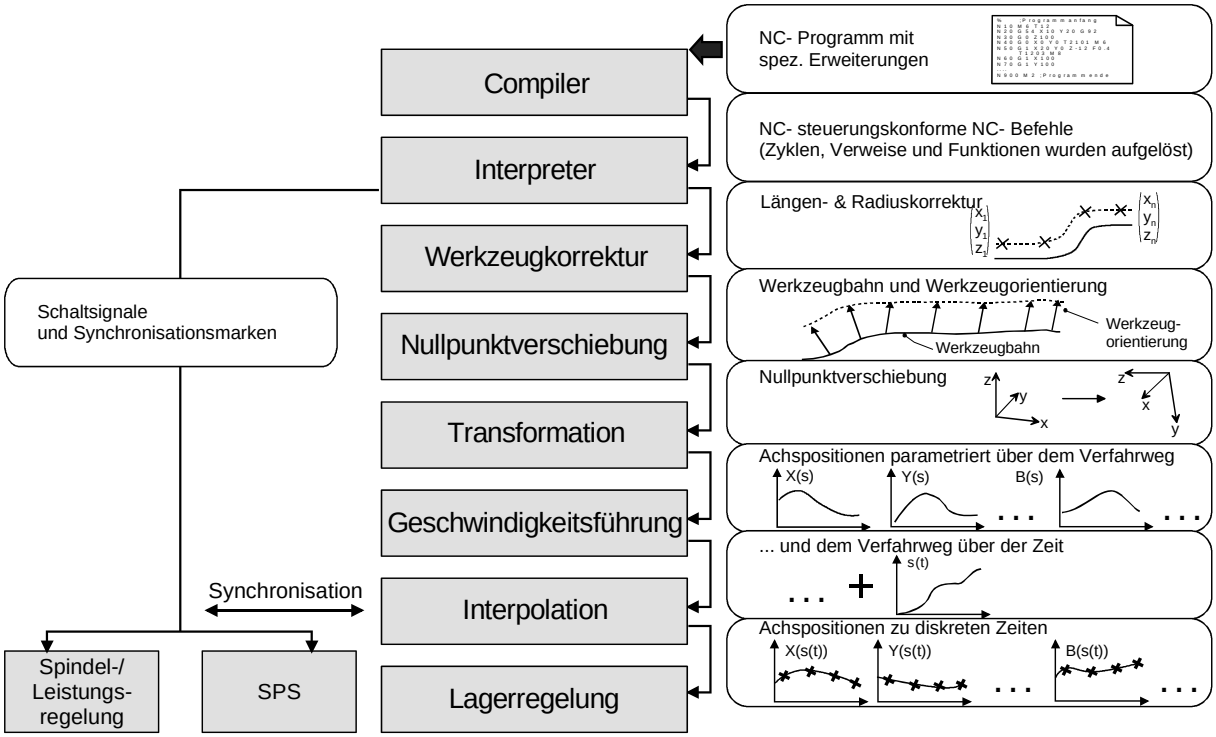


Bild 2.7: NC-Programmverarbeitung durch die Teilfunktionen einer NC-Steuerung
Processing of the NC-program by sub-tasks of a numerical control

NC-Interpreter

Die nächsten Schritte werden automatisch durch den Programmstart ausgelöst. Der Interpreter der Steuerung ruft den Parser auf, der die alphanumerischen Zeichen des vom Compiler vorübersetzten NC-Programms einliest, sie auf der Grundlage der steuerungsspezifischen Programmierschnittstelle analysiert und sie den internen softwaretechnischen Funktions- und Datenstrukturen der NC-Steuerung zuweist. Je nach Steuerungsaufbau werden dadurch Funktionsketten aufgerufen [BOSC01] oder objektorientierte Datenstrukturen initialisiert [SIEM00]. Diesen Vorgang wiederholt der Parser für alle aufeinander folgenden Programmzeilen, wobei er Sprunganweisungen und Unterprogramme erkennt und auflöst. Da der Parser schneller lesen kann, als die Steuerung die Anweisungen umsetzt und abarbeitet, entsteht ein Vorlauf zur Programmverarbeitung. Die Größe dieses Vorlaufs wird durch die Speicherverwaltung der Steuerung kontrolliert. Er umfasst bei modernen NC-Steuerungen mehrere 100 NC-Sätze. Dieser Vorlauf ist unter anderem notwendig, um einen kontinuierlichen Datenfluss sicherzustellen und um Werkzeugkorrekturen, Variablen oder Geschwindigkeitsprofile satzübergreifend verrechnen zu können.

Des Weiteren übernimmt der Interpreter die Aufgabe, Programmstrukturen und Befehle aufzulösen, die sich nicht unmittelbar auf die Maschinenfunktionen und Verfahrbefehle der nachfolgenden Steuerungsmodule abbilden lassen. Ein typisches Beispiel sind Subroutinen und NC-Zyklen. Bei der Auflösung der darin kodierten Merkmals- und Operationsdefinitionen übernimmt der Interpreter die Funktion eines Postprozessors. Er löst bei Bedarf einen Werkzeugwechsel aus, berechnet Anfahr-, Zustell-, Bearbeitungs- und Abfahrbewegungen und generiert die notwendigen Technologieparameter sowie Schaltfunktionen. Das technologische Wissen der Programmauflösung ist Eigentum des Steuerungsherstellers oder wird anwenderspezifisch als Zyklus entwickelt und in die Steuerung integriert. Die Auflösung eines Merk-

mals variiert je nach Hersteller und ist für den Anwender nicht immer nachvollziehbar, wie beispielsweise die verschiedenen Algorithmen zur Auflösung von Taschenproblemen.

Neben der Programmauflösung und dem Aufruf interner Steuerungsfunktionen hat der Interpreter die Aufgabe, die Abarbeitungsreihenfolge der generierten internen Anweisungen zu synchronisieren. Er erkennt sequentiell abzuarbeitende, parallele und modal, d.h. übergreifend gültige Befehle, und steuert über die Weiterleitung der Befehle und Wartemarken ihre funktions-, positions- und zeitsynchrone Ausführung in der SPS und an den Maschinenachsen.

Geometrieverarbeitung

Der Interpreter hat die Programmanweisungen in steuerungsinterne Schalt- und Geometrieanweisungen aufgelöst und um Marken zur Synchronisation ihrer weiteren Abarbeitung ergänzt. Während die Schaltanweisungen an die Befehlspuffer der SPS geleitet werden, müssen die Geometrieinformationen zur Anpassung an Werkzeug, Spannlage und Maschinenkinematik weiterverarbeitet werden.

In der Regel wird in einem NC-Programm die zu fertigende Werkstückkontur programmiert und nicht der Bezugspunkt des Werkzeugs, bzw. der maschinenspezifische Anlenkpunkt der Maschinenachsen. Die Werkzeugkorrektur hat daher die Aufgabe, diesen werkzeugneutralen Bezugspunkt zu berechnen, indem die programmierte Bahn äquidistant um die Werkzeugabmaße verschoben wird. Dabei entstehen neue Geraden- und Kreissegmente, die miteinander zu längeren Bahnsegmenten verschmolzen und je nach Steuerung in ein höherwertiges Kurvenformat, z.B. Spline, umgewandelt werden [KLEI95, SIEM04a]. Dabei geht die direkte Zuordnung zwischen den im NC-Programm kodierten Bahnsegmenten und den steuerungsintern weiterzuverarbeitenden Kurvenformaten verloren.

Die Nullpunktverschiebungen werden bereits in der Werkzeugkorrektur verrechnet oder in der nachfolgenden Transformation berücksichtigt. Sie sind notwendig, um die reale Aufspannlage und geänderte Aufmaße des Werkstücks kompensieren zu können, ohne das NC-Programm neu erstellen zu müssen.

In der Transformation werden die kartesischen Positions- und Wegangaben auf die Kinematik der Maschine abgebildet. Je nach Maschinenkinematik ist die Transformation nicht eindeutig und lässt sich nicht mathematisch geschlossen lösen. In diesem Fall werden numerische Algorithmen eingesetzt, die die Verfahrwege interpolieren und punktweise transformieren. Dadurch kann es zu geringfügigen, in den Steuerungsparametern tolerierten Bahnabweichungen kommen [WECK01d].

Ausgehend von den im NC-Programm vorgegebenen Technologieparametern und den dynamischen Grenzen der Maschinenachsen berechnet die Geschwindigkeitsführung ein Geschwindigkeitsprofil über dem Verfahrweg. Dabei wird die Bahngeschwindigkeit teilweise bis auf Null reduziert. Dies ist notwendig, um beispielsweise Bauteilkanten exakt zu umfahren oder aufgrund der kinematikbedingten Richtungsumkehr einer Maschinenachse [FAUS97]. Daraus folgt, dass die Geschwindigkeitsprofile in Abhängigkeit von der Aufspannlage variieren, je nachdem ob durch die Spannlagenkorrektur die Bewegung in die Richtung einer Achse mit höheren Beschleunigungspotentialen transformiert wird.

Interpolation und Lageregelung

Der Interpolator berechnet in Abhängigkeit von dem Geschwindigkeitsprofil für die einzelnen Maschinenachsen Positionssollwerte zu diskreten Zeiten. Wenn nicht bereits durch den Interpreter gesteuert, erfolgt an dieser Stelle die Synchronisation zwischen der Bewegungsausführung, der Arbeitsspindel und den in der SPS auszuführenden Schaltfunktionen. Danach übernimmt die Antriebsregelung (Lage-, Strom- und Geschwindigkeitsregler) die Ansteuerung der Antriebe, so dass der Sollverlauf der Achsen trotz der wirkenden Prozesskräfte und der mechanischen Ungenauigkeiten des Antriebsstrangs eingehalten wird. Bei digitalen Antrieben

haben die Achsen eigene Regelkarten mit Datenspeichern, in denen die interpolierten Sollwerte mit Vorlauf zwischengespeichert werden. Auf die Verarbeitung in den Reglern hat die NC-Steuerung keinen direkten Zugriff. Auch sind die Antriebsregler i.d.R. nicht zur Funktionserweiterung offen gelegt. Bosch Rexroth bietet als erster Hersteller eine offene Schnittstelle im Antrieb für einfache SPS Funktionen.

Infolge der Konvertierungen und Transformationen innerhalb der Geometriekette sowie des mechanischen Verhaltens der Antriebssysteme kann ein NC-Programm nur mit einer begrenzten Genauigkeit steuerungstechnisch ausgeführt werden. Der programmierte Start- und Endpunkt eines NC-Wegsements muss nicht zwingend mit den an den Lageregler zur Ausführung übergebenen Sollwerten übereinstimmen. Zudem gibt es in der Programmausführung zeitliche Schwankungen, die durch das veränderliche Geschwindigkeitsprofil, die Synchronisation mit den zu schaltenden Maschinenfunktionen sowie die Abarbeitungsgeschwindigkeit der verschiedenen Datenpuffer bedingt sind.

2.2.2. Aufbau einer NC-Steuerung

Architecture of a numerical control

Moderne NC-Steuerungen bestehen aus einer oder mehreren Rechnereinheiten, auf denen verschiedene Softwareapplikationen installiert sind, die die Steuerungsfunktionalität enthalten. Die Anzeige- und Eingabegeräte, die Antriebe sowie die Anpasssteuerung der Werkzeugmaschine sind über spezielle, oftmals steuerungsspezifische Datenschnittstellen mit der NC-Steuerung verbunden. An diese sogenannten Antriebsschnittstellen und Bussysteme werden besondere Anforderungen gestellt. Sie müssen die Steuerungsinformationen synchron, deterministisch, d.h. in einer vordefinierten Zeitspanne, und verlustfrei weiterleiten, um eine schnelle und fehlerfreie Maschinenansteuerung sicherzustellen.

Je nach Steuerung werden unterschiedliche Rechnersysteme eingesetzt. Es gibt Steuerungen, in denen laufen die Benutzerschnittstelle und die Grundsteuerung, d.h. der NC-Kern, auf einem Industrie PC (z.B. Beckhoff NC). Andere Hersteller setzen zwei getrennte PCs ein (z.B. Siemens *Sinumerik 840D*) oder lassen den Kern auf einer in den Steuerungs-PC integrierten Einsteckkarte mit eigenem Prozessor und Betriebssystem (z.B. Bosch Rexroth *MTX*) laufen. In Abhängigkeit von der gewählten Konfiguration sind die Funktionsaufrufe und der Datenaustausch auf dem gemeinsamen Arbeitsspeicher realisiert, als so genanntes Shared Memory, oder sie müssen über Schnittstellen, wie Ethernet oder Profi-Bus, in Form von Nachrichten, d.h. eigens erzeugten Datenpaketen, übertragen werden. Der letztgenannte Fall ist aufgrund von Datenkonvertierungen und der physikalischen Übertragung langsamer. In Abhängigkeit von dem Übertragungsprotokoll entstehen Verzögerungen von einer (z.B. SERCOS Antriebsschnittstelle) bis zu mehreren Millisekunden (SPS) [SAUE04]. Dieser Nachteil wird akzeptiert, um die Teilsysteme wie die Benutzeroberfläche, Programm- und Geometrieverarbeitung, SPS sowie die Antriebe dezentralisieren zu können.

Ferner unterscheiden sich die NC-Steuerungen im Aufbau und in der Funktion ihrer Software. Auf der Basis moderner Programmiersprachen, wie C oder C++, sind funktions- und datenorientierte Steuerungsarchitekturen entstanden. In den funktionsorientierten, prozeduralen Systemen werden extern gesteuert Funktionen aufgerufen, die die eingelesenen NC-Programmanweisungen schrittweise weiterverarbeiten. Dagegen generieren die objektorientierten, modularen Systeme Daten als Eingangsinformation für die nachfolgenden Module. Die Datenverarbeitung erfolgt über spezialisierte Methoden der jeweiligen Objekte [HARD02].

Die Umsetzung des NC-Programms ist abhängig vom Steuerungshersteller. Folglich sind die internen Datenformate, Kommunikationsschnittstellen und Funktionen der NC-Programmverarbeitung nicht normiert. Die Möglichkeiten der Nutzerinteraktion beschränken sich auf die Benutzeroberfläche, mit den darin integrierten NC-Programmiermöglichkeiten und auf

Override Schalter zur Geschwindigkeits- und Leistungsregelung sowie auf Regeleinrichtungen, die am Signalausgang der NC-Steuerung, d.h. auf der Ebene der Sollwerte eingreifen. Die Programmverarbeitung kann der Anwender nicht beeinflussen.

2.2.3. Offene NC-Steuerungen

Open numerical controls

NC-Steuerungen werden in zunehmendem Maße nicht mehr speziell für eine Technologie und Maschine entwickelt, sondern als universelle Steuerungen mit diversen Erweiterungsoptionen angeboten. Die Aufgabe, die Steuerungsfunktionalität an die individuellen Anforderungen der Maschine, des Prozesses und des Anwenders anzupassen, obliegt den Maschinen- oder Systemherstellern. Auf der Basis einer softwarebasierten CNC-Logik kann das spezifische Technologie- und Prozesswissen in die Steuerung integriert werden, ohne dass Änderungen am Hardwaresystem vorgenommen werden müssen. Hierzu sind jedoch offene Schnittstellen zur Basissoftware der NC-Steuerung notwendig [BITZ96].

Jede NC-Steuerung bietet Schnittstellen an, um beispielsweise das NC-Programm einzulesen, den Steuerungskern zu parametrieren und die externen Antriebe sowie Maschinenfunktionen anzusteuern. Des Weiteren passt der Steuerungs- oder Maschinenhersteller über diese Schnittstellen die NC-Steuerung an die kinematischen und funktionellen Unterschiede der einzelnen Werkzeugmaschinen an. Eine reine Schnittstellenkonfiguration und Funktionsparametrierung reicht jedoch nicht aus, um prozess- und anwenderspezifische Anforderungen, funktionelle Weiterentwicklungen sowie die Anpassung an Sondermaschinen zu realisieren.

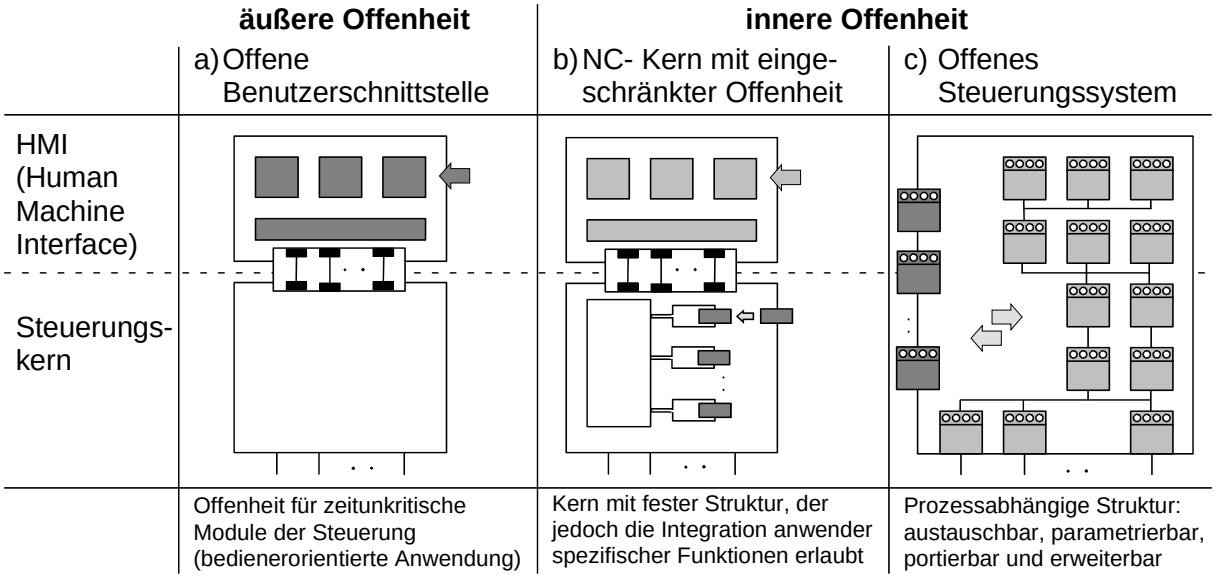

äußere Offenheit
innere Offenheit
a) Offene Benutzerschnittstelle
b) NC- Kern mit eingeschränkter Offenheit
c) Offenes Steuerungssystem
HMI (Human Machine Interface)
Steuerungs-kern
Offenheit für zeitunkritische Module der Steuerung (bedienerorientierte Anwendung)
Kern mit fester Struktur, der jedoch die Integration anwender spezifischer Funktionen erlaubt
Prozessabhängige Struktur: austauschbar, parametrierbar, portierbar und erweiterbar

Bild 2.8: Offenheit einer NC-Steuerung nach [WECK96]

Openness of a numerical control

Eine Voraussetzung für die Erweiterung des Funktionsumfangs ist eine offene NC-Steuerung [PRIT93]. Es wird zwischen äußerer und innerer Offenheit unterschieden. Die äußere, anwenderorientierte Offenheit umfasst eine harmonisierte NC- und SPS-Programmierung, eine adaptierbare Benutzerschnittstelle (Bild 2.8, a) sowie standardisierte Kommunikations- und Antriebsschnittstellen. Die innere Offenheit bezieht sich auf den Steuerungskern. Sie fordert weitreichende Zugriffsmöglichkeiten auf Steuerungs- und Prozessdaten, konfigurierbare Software mit offenen Schnittstellen (Bild 2.8, b, c) sowie eine modulare aufgebaute Hardware [WECK93a, OSAC00].

2.2.3.1. Industrielle Anwendung offener Steuerungssysteme

Industrial application of open numerical controls

Die PC- und Windows-basierten Benutzerschnittstellen moderner NC-Steuerungen erfüllen in der Regel die Anforderungen nach äußerer Offenheit. Es ist möglich, Fremdapplikationen in die Steuerungsoberfläche zu integrieren, diese mit den Softkeys der Steuerung zu bedienen und über standardisierte Schnittstellen, wie DDE (Dynamic Data Exchange) oder OPC (OLE for Process Control), auf Steuerungsdatenbanken und Steuerungssignale zuzugreifen. Typische Anwendungen sind Applikationen zur Werkzeugverwaltung, die Anzeige von Maschinenzuständen und die Fehlerdiagnose. Die Maschinenhersteller nutzen diese Offenheit zur Entwicklung eigener, maschinen- und prozessorientierter Benutzeroberflächen [REIM04].

Auf der Ebene der NC-Programmierschnittstelle bieten die Steuerungshersteller die Möglichkeit, die Befehlsworte zu konfigurieren und entsprechend den Maschinenfunktionen zu erweitern. Darüber hinaus lassen sich in Steuerungen, wie der *MTC* von Bosch Rexroth, Compiler integrieren, über die spezifische NC-Programmerweiterungen auf die NC-Programmierschnittstelle der Steuerung „vorübersetzt“, d.h. abgebildet werden [INDR96] (Bild 2.9).

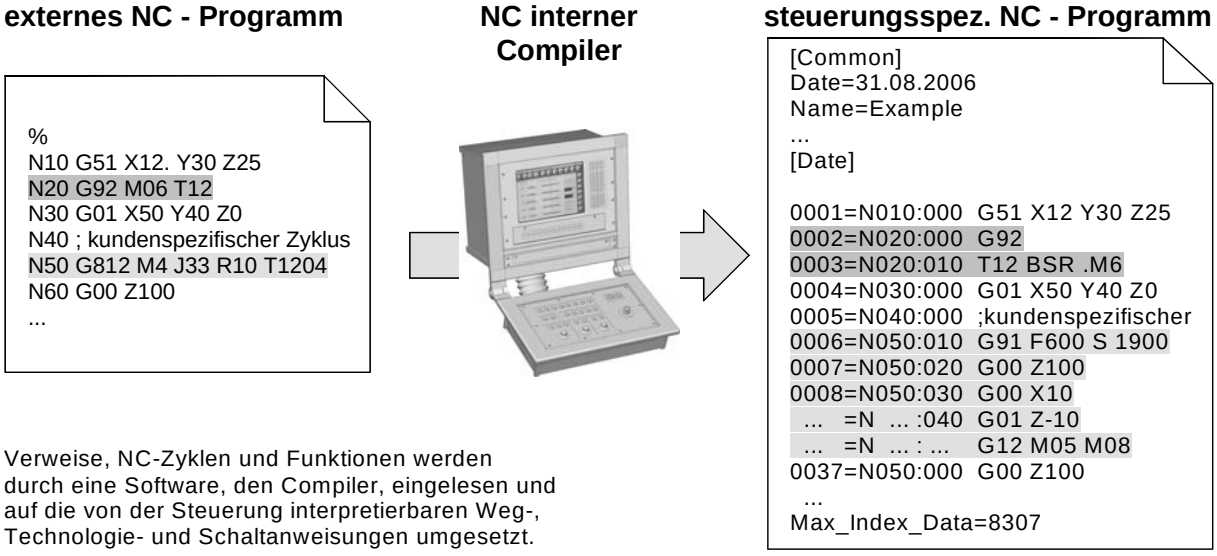


Bild 2.9: Vorverarbeitung des NC-Programms mit Compilern

Preprocessing of the NC-program with compilers

Eine herstellerübergreifende Offenheit des NC-Kerns, wie ihn z.B. die OSACA Spezifikation vorschlägt, wurde von den Steuerungsherstellern nicht realisiert [OSAC00, BRÜH03]. Stattdessen werden einzelne, proprietäre Lösungen zur Integration echtzeitfähiger Anwendungen in den NC-Kern angeboten. An dieser Stelle werden exemplarisch die offenen Schnittstellen zur Steuerungserweiterung der *Sinumerik 840D* (Siemens) und der Bosch Typ 3 osa bzw. ihrer Nachfolgesteuerung der *MTX* (Bosch Rexroth) vorgestellt.

Offenheit der Siemens Sinumerik 840D

Der Steuerungshersteller Siemens bietet für die *Sinumerik 840D* eine proprietäre Entwicklungsumgebung an, um anwenderspezifische Funktionen, sogenannte Compile-Zyklen zu entwickeln und in den vorgegebenen Funktionsablauf des Siemens Kerns einzubinden (Bild 2.10). Der Kern der *Sinumerik 840D* ist objektorientiert in C++ programmiert. Seine Struktur wird durch die logischen Funktionen der Steuerung in Form von Modulen vorgegeben. Jedes der Module stellt eine in sich geschlossene Applikationen dar, die nicht manipuliert oder ausgetauscht werden kann. Beim Abarbeiten des NC-Programms werden diese Module in einer definierten Abfolge, die der Entwickler nicht beeinflussen kann, sequentiell durchlaufen. An-

hand von fest implementierten Aussprungfunktionen, den sogenannten Events, kontrolliert der Steuerungshersteller die Stellen im Ablauf, an denen kundenspezifische Erweiterungen in die Programmverarbeitung eingefügt werden können. Aus der Sicht des Entwicklers spiegeln die Events den Steuerungsaufbau wider und stellen für ihn die offene Funktionsschnittstelle zum NC-Kern dar. Die mehr als 50 verfügbaren Events erlauben eine Manipulation der Programmverarbeitung beginnend beim Einlesen des NC-Programms über die Geometrieverarbeitung und die Interpolation bis in die Lageregelung. Den offenen Zugriff auf die Daten der Steuerung kontrollieren Bindings. Ein Binding ist eine vordefinierte, in C++ implementierte Zugriffsmethode, die der Entwickler zum Schreiben und Lesen von Systemdaten oder zum Abfragen von Zustandswerten nutzen kann. Darüber hinaus bietet die Entwicklungsplattform der *Sinumerik 840D* die Möglichkeit, eigene anwenderspezifische Daten direkt zu verwalten [SIEM03a].

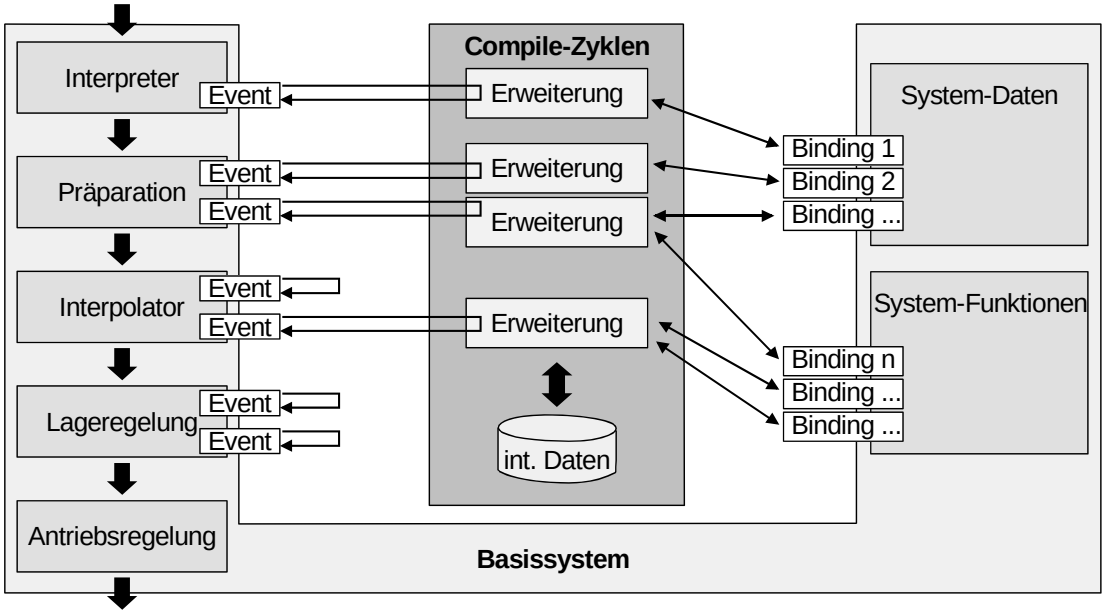


Bild 2.10: Objektorientierter, offener NC-Kern der *Sinumerik 840D* (nach [SIEM03a])
Object oriented, open NC-Kernel of the Sinumerik 840D

Offenheit der Bosch Typ3 osa / MTX

Im Gegensatz zu den Compilezyklen der *Sinumerik 840D* bietet die *Bosch MTX*, eine weiter reichende Offenheit zum NC-Kern. Dazu wird dem Entwickler neben Schnittstellen zur Integration eigener Funktionen die offene Konfiguration der Funktionsabläufe und damit Einfluss auf die Struktur des Steuerungskerns ermöglicht.

Der Kern der *MTX* setzt sich aus einer Reihe von Subsystemen zusammen. Jedes dieser Subsysteme stellt eine geschlossene Teilfunktion der Steuerung dar und beinhaltet mehrere Einzelfunktionen und Prozeduren, die sogenannten Jobs. Beispiele für Subsysteme sind die Satzvorbereitung, die Bewegungserzeugung, die Werkzeugkorrektur und die Interpolation.

Der Anwendungsentwickler implementiert seine Funktionen als eigenständige Jobs in der Programmiersprache C und fügt sie einem Subsystem der Steuerung zu. Für jede NC-Funktion, d.h. einen NC-Programmbefehl, werden in Form einer Liste die auszuführenden Jobs konfiguriert. Die Funktionalität einer NC-Funktion ist folglich durch die Summe der ihr zugeordneten Funktionen und deren Abfolge beschrieben und lässt sich vom Entwickler frei beeinflussen. Beispielsweise kann sich die Abarbeitung eines linearen Verfahrbefehls aus Jobs zum Einlesen des Befehls, zum Bestimmen der Start- und Endposition, der kinematischen Transformation, der Geschwindigkeitsführung etc. zusammensetzen.

Da die *MTX* nicht objektorientiert, sondern strukturiert in Form von Prozeduren und Funktionen aufgebaut ist, lassen sich ihre internen Daten nicht als Objekte kapseln. Der Datenzugriff wird daher zentral über die NC-Schnittstelle (NCS) gesteuert (Bild 2.11). In der NCS wird zwischen modalen Steuerungsdaten, auf die systemweit zugegriffen werden kann, globalen Daten innerhalb eines Subsystems, lokalen Daten für die Jobs einer NC-Funktion und privaten Daten, die ausschließlich durch den einzelnen Job genutzt werden können, unterschieden [BOSC96, BOSC01].

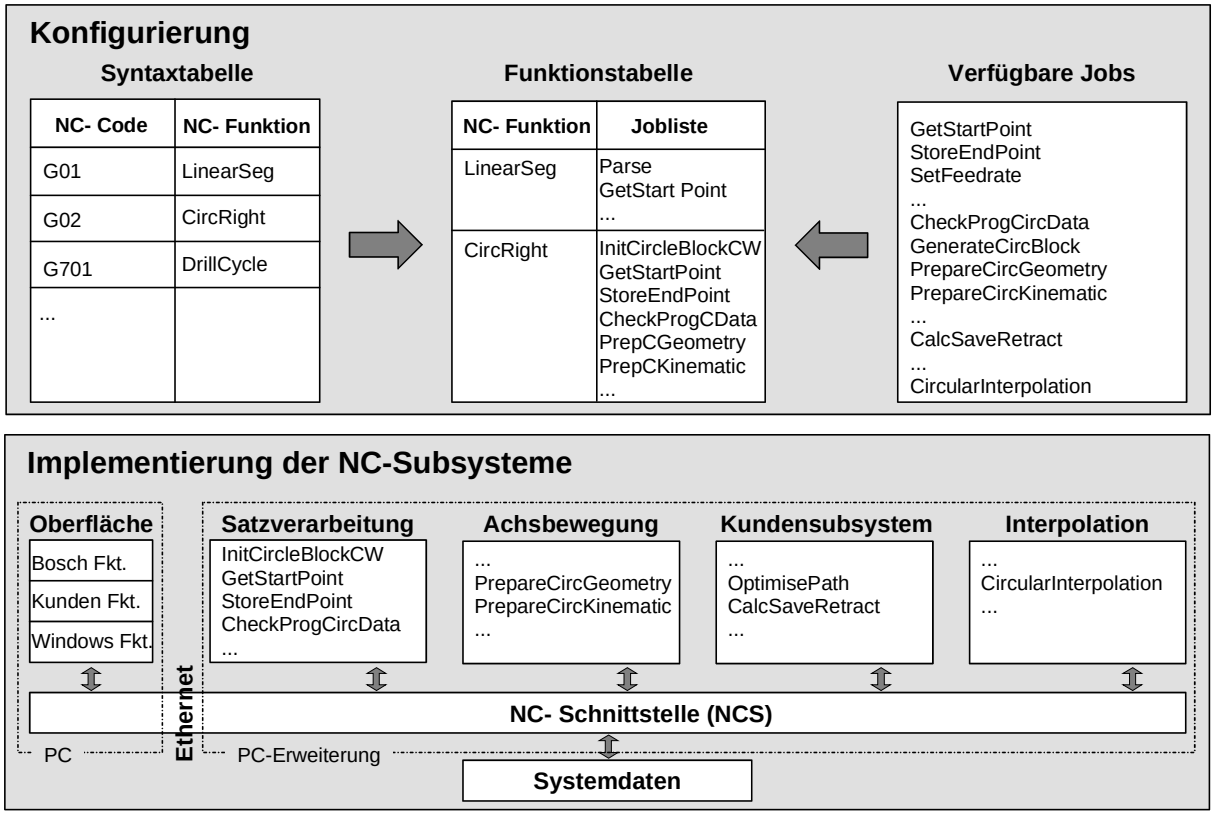


Bild 2.11: Prozeduraler, offener NC-Kern der Bosch Typ 3 osa / MTX
Procudural, open NC-Kernel of the Bosch Typ 3 osa / MTX

2.2.3.2. Offenheit in der NC-Programmumsetzung

Openness in processing the NC-commands

Die Offenheit industriell angebotener Systeme, als auch die steuerungsübergreifender Initiativen orientieren sich primär an den Steuerungsfunktionen und einer sequentiellen NC-Programmverarbeitung. Im Fokus stehen der Austausch von Funktionsmodulen, wie beispielsweise der Benutzerschnittstelle, der Achstransformation oder eine Überwachung und Manipulation der Sollwerte. Die Datenschnittstellen zwischen den Modulen sind weitestgehend vorgegeben. Sie sind auf die Funktionen der Teilmodule ausgerichtet und nicht auf den Austausch zusätzlicher Daten, wie beispielsweise das Bearbeitungsfeature, die operativen Technologie-, Strategie- und Werkzeugvorgaben oder Parameter für eine Prozessüberwachung und -regelung.

Daher werden die mnemonischen Codes der Programmierschnittstelle mithilfe von Compilern oder im Interpreter auf die elementaren, internen Steuerungsfunktionen heruntergebrochen. Eine durchgängige Nutzung der NC-Programmvorgaben bis in die Sollwertebene, um neben den Weg- und Schaltanweisungen Prozessparameter, Grenzwerte und Regelstrategien auszutauschen, ist nicht vorgesehen. Ein Taschenzyklus muss daher nach Programm abgearbeitet

werden und der Anwender kann nicht zwischen den einzelnen Schnittebenen oder innerhalb einer Ebene die Zustellung, das Werkzeug oder die Bahnstrategie ändern. Dafür müsste die NC-Steuerung neben einer Referenzarchitektur und offenen, funktionsorientierten Schnittstellen eine durchgängige Datenbasis des NC-Programms und eine rückgekoppelte Programmumsetzung unterstützen. Als Beispiel für eine offene Referenzarchitektur sei auf die im Rahmen von OSACA erarbeiten Standards verwiesen: [OSAC00, HARD02, BRUE03, KAHM05]

2.3. Prozessüberwachung und –regelung

Process monitoring and control

Auch in hochautomatisierten Bearbeitungsabläufen treten Prozessschwankungen, Störungen und Fehler auf. Im Fall der NC-Bearbeitung reichen die Folgen von Qualitätsschwankungen am Werkstück, erhöhtem Werkzeugverschleiß, einer suboptimalen Produktivität bis hin zu Beschädigungen aufgrund von Kollisionen am Werkstück, Werkzeug oder an der Maschine. Bezogen auf das NC-Programm und dessen Abarbeitung lassen sich drei wesentliche Ursachen unterscheiden:

- menschliche Fehler beim Programmieren und Rüsten
- Abweichungen zwischen dem angenommenen und dem realen Prozessverlauf
- externe Einflüsse, wie Werkzeugverschleiß, Maschinenverhalten, Störungen

Mit dem Ziel, die Prozesssicherheit, Fertigungsqualität und Produktivität numerisch gesteuerter Maschinen zu erhöhen, setzt die Industrie daher Systeme zur Prozessüberwachung und Regelung ein [KÖNI02]. Dabei ist es die Aufgabe der Prozessüberwachung, Störungen automatisch zu erkennen und geeignete Reaktionen in Form von Meldungen, Abschalt- oder Korrekturbefehlen auszulösen. Dagegen hat die Prozessregelung das Ziel, unter fortlaufender, selbstständiger Anpassung der Prozessparameter an das Verhalten von Maschine, Werkzeug, Werkstück und Material eine optimale Prozessführung zu erreichen. Die VDI-Richtlinie 3426 unterteilt die adaptiven Regelsysteme aufgrund ihrer Zielsetzung in Grenzwert- (Adaptive Control Constraint, ACC) und Optimierregelungen (Adaptive Control Optimization, ACO). Ferner werden technologische, d.h. nach wirtschaftlichen Gesichtspunkten arbeitende, und geometrische, d.h. auf die Einhaltung der tolerierten Geometriemerkmale ausgerichtete Regelsysteme unterschieden [VDI3426].

2.3.1. Anwendungen zur Prozessregelung

Applications for process control

Seit den 60er Jahren werden Konzepte zur Prozessregelung entwickelt und an numerisch gesteuerten Werkzeugmaschinen eingesetzt. Im Fokus der Entwicklung standen anfangs Systeme zur Überwachung von Überlasten und zur Regelung technologischer Parameter. Mit ACC Systemen regelte man beispielsweise den Bahnvorschub oder die Spindeldrehzahl, um die Zerspanleistung zu maximieren [MAIE74]. Andere Konzepte sahen zusätzlich eine Manipulation der Schnittaufteilung vor. *Gieseke* und *Breuer* stellten derartige Systeme für die Drehbearbeitung und *Gather* für das Stirnfräsen vor [GIES73, GATH77, BREU83]. Die industrielle Akzeptanz und der Funktionsumfang dieser Systeme wurden jedoch durch den hohen Aufwand, die verfügbare Sensorik, die Leistungsfähigkeit der Rechnersysteme und die fehlenden Schnittstellen zur NC-Steuerung begrenzt, so dass viele Konzepte erst in der jüngeren Zeit in der Fertigung umgesetzt werden können [VITR06].

Aus der Sicht der Systemhersteller vereinfachen die modernen NC-Steuerungen mit ihren offenen Kommunikations- und Softwareschnittstellen sowie den steuerungs- und antriebsinternen Sensoren die Entwicklung und die steuerungsübergreifende Integration von Überwa-

chungs- und Regelungssystemen [LANG05]. Beispiele sind das Brankamp-Artis-Prometec-Standard-Interface BAPSI zur Anbindung von Überwachungssystemen an die SPS [BAPS97], die digitale SERCOS Antriebsschnittstelle zum standardisierten Zugriff auf Antriebsdaten [SERC02] oder die herstellerspezifischen Entwicklungsumgebungen zur NC-Kern-Erweiterung [BOSC04, SIEM03a]. In der Tabelle 2.1 sind exemplarisch industriell eingesetzten Systeme zur Prozessüberwachung und -regelung für verschiedene numerisch gesteuerte Fertigungsverfahren zusammengefasst.

	Laserbearb.	Bohren	Drehen	EDM	Fräsen	Schleifen
Prozessüberwachung						
Anschnitterkennung		x			x	**X**
Werkzeugbruch		**X**	**X**	**X** (Draht EDM)	**X**	**X**
Werkzeugverschleiß		**X**	**X**		**X**	**X**
Prozessregelung: a) Parameter						
Drehzahlregelung		x	x		x	
Vorschubregelung		x		**X** (Senk./Draht.)	x	**X**
Schnittkraftregelung		–	–		–	
Fokuslagenregelung	**X**					
Leistungsregelung	**X**					
Prozessregelung: b) Verfahrwege						
Zustellung			–	– (Bahn-EDM)	–	x (=Vorschub)
Schnittaufteilung		– (Spanleeren)	–	– (Bahn-EDM)		
Nahtverfolgung	**X**					

X Stand der Technik
x vereinzelte Anwendungen
– industriell nicht eingesetzt

Tabelle 2.1: Aktuelle Anwendungen zur Prozessüberwachung und –regelung
Applications for process monitoring and control

Der Überblick zeigt, dass beim Bohren, Drehen und Fräsen die Systeme zur Werkzeugverschleiß- und -bruchüberwachung im Fokus der aktuellen Anwendungen stehen. Lösungen zur adaptiven Regelung des Vorschubs oder der Schnittgeschwindigkeit sind dagegen seltener. Anwendungen mit einer prozessabhängigen Schnittaufteilung sind nicht im industriellen Einsatz zu finden [WECK01c, KAHM05]. Im Folgenden werden einzelne Applikationsfälle exemplarisch erläutert.

Anschnitterkennung

In der Schleifbearbeitung gehört die Anschnitterkennung zum Stand der Technik. Sie wird eingesetzt, um während den sicherheitsbedingten Luftschnitten mit höheren Geschwindigkeiten zustellen zu können und so die unproduktiven Nebenzeiten zu minimieren. Den Kontakt mit dem Werkstück detektieren die Systeme über Körperschallmessungen und ein Ansteigen der Vorschubkraft [NORD04a].

Werkzeugüberwachung

In der automatisierten Serienfertigung, als auch in schnell ablaufenden Prozessen werden Systeme zur Werkzeugüberwachung eingesetzt, um beim Erreichen von Verschleißkriterien das Werkzeug zu wechseln oder die Bearbeitung nach einem Werkzeugbruch sicher zu stoppen, bevor größere Schäden entstehen. Im Rahmen dieser Arbeit spielt die Bruchüberwachung

jedoch eine untergeordnete Rolle, da sie zu einem Abbruch der Bearbeitung führt und es sich folglich nicht um die Regelung des Bearbeitungsprozesses handelt.

Dagegen ist der Werkzeugverschleiß ein bearbeitungsparalleler, kontinuierlicher Vorgang, der negativen Einfluss auf die Bauteilqualität und die Produktivität hat. Bei den NC-Steuerungen gehört es daher zum Stand der Technik, die Werkzeugstandzeit als Kenngröße für den Werkzeugverschleiß zu überwachen. Hierzu werden intern die zeitliche Einsatzdauer oder die Anzahl der bearbeiteten Werkstücke protokolliert. Beim Überschreiten der vom Anwender in einer Werkzeugdatenbank vorgegebenen Kriterien sperrt die Steuerung das Werkzeug für die nachfolgenden Bearbeitungsoperationen. Beim nächsten Aufruf des verschlissenen Werkzeugs wird automatisch ein identisches Schwesterwerkzeug eingewechselt oder die Bearbeitung unterbrochen und der Werker zum Wechsel aufgefordert. Ein direkter Werkzeugwechsel im Prozess ist dagegen nicht vorgesehen [INDR97, SIEM04]. Ein Grund hierfür sind fehlende Programminformationen zur Bearbeitungsgeometrie. Die NC-Steuerung kann Ab- und Anfahrstrategien nicht autonom ergänzen, um in der Bearbeitung an beliebiger Stelle abzuheben und mit einem neuen Werkzeug wieder zuzustellen, ohne Oberflächenfehler zu erzeugen.

Neben diesen statistischen Lösungen werden auch Systeme zur sensorbasierten Werkzeugverschleißüberwachung eingesetzt. Typische Messgrößen sind die Schnittkraft, das Antriebsmoment oder der Antriebsstrom der Werkzeugspindel. Die Signalwerterfassung basiert teilweise auf steuerungsinternen Antriebssignalen, überwiegend jedoch auf externer Sensorik [KAEV04]. Die Auswertung der gemessenen Signale erfolgt entweder gegen vorgegebene Grenzwerte, mitlaufende Bandbreiten oder eingelernte Vergleichsmessungen [ARTI00, PROM00, NORD04]. Da es sich bei den Systemen vorrangig um Überwachungssysteme handelt, greifen sie nicht regelnd in den Bearbeitungsprozess ein, sondern zeigen den Werkzeugzustand in der Benutzerschnittstelle der Steuerung an, sperren das Werkzeug in der Werkzeugverwaltung oder unterbrechen die Bearbeitung.

Kollisionsüberwachung

Wie bereits im Unterkapitel zur NC-Programmierung dargestellt, ist die Überprüfung auf Kollisionen eine wichtige Teilaufgabe der Simulation und des Einfahrens, um Schäden und Ausfallzeiten an der Maschine zu vermeiden. Aufgrund von fehlenden Simulationsmöglichkeiten, Programmierfehlern, abweichenden Rüstsituationen an der Maschine und vor Ort neu einzufügenden Verfahrbewegungen beim Wiederanfahren nach einem Programmabbruch lassen sich Kollisionen jedoch nicht gänzlich ausschließen. Dem Anwender werden daher sensorbasierte und softwaretechnische Überwachungssysteme zum Erkennen von Kollisionen im Bearbeitungsprozess angeboten.

Die sensorbasierte Kollisionsüberwachung arbeitet analog zu den Systemen zur Werkzeugüberwachung. Sie überwacht die auf das Werkzeug oder das Werkstück wirkenden Kräfte und Momente. Kommt es zu einem sprunghaften Anstieg der Kräfte oder werden die vorgegebenen Grenzwerte und eingelernten Kraftverläufe überschritten, wird die Bearbeitung abgebrochen und die Maschine schnellstmöglich gestoppt [NCFE03]. Ein Abgleich der Messwerte mit den im NC-Programm kodierten Verfahrwegen und Prozessparametern findet nicht statt. Einerseits fehlen im NC-Programm Informationen über das Rohteil und damit über den theoretischen Zeitpunkt und Ort des Kontakts zwischen Werkzeug und Werkstück und andererseits gibt das NC-Programm keine Auskunft über die zu erwartenden Eingriffsbedingungen.

Darüber hinaus bieten verschiedene Steuerungshersteller eine softwaretechnische Kollisionsüberwachung in Form einer Einschränkung und Überwachung der Achsverfahrbereiche an. Dazu werden für die Maschine, die Werkzeuge und Spannelemente Schutzräume definiert, die sich nicht überschneiden dürfen, bzw. in die das Werkzeug nicht einfahren darf. Je nach System wird der Anwender gewarnt, der Verfahrbefehl zur Quittierung unterbrochen oder die Achsbewegung abgebrochen [NCFE05]. Die Verfahrweggrenzen und Schutzräume werden entweder in den Maschinendaten, durch steuerungsspezifische NC-Programmbefehle oder in

separaten Datenfiles vorgegeben. Eine direkte Übernahme aus dem CAP/CAM oder eine standardisierte Parametrierung im NC-Programm ist heute nicht möglich.

Adaptive Vorschubregelung

Grenzwertregelungen (ACC Systeme) setzt die Industrie ein, um beispielsweise während der Schruppbearbeitung die Vorschubgeschwindigkeit in Abhängigkeit vom Werkzeugverschleiß zu maximieren. Damit umgeht sie die auf das Standzeitende der Werkzeuge ausgelegten, mit Sicherheitszuschlägen abgeschätzten und daher suboptimalen Vorgaben des NC-Programms [SAUR03]. Die für die Fräsbearbeitung angebotenen Systeme basieren auf einer Überwachung der Leistungsaufnahme oder des Drehmoments an der Arbeitsspindel. Dazu vergleichen sie die gemessenen Werte mit zuvor im Regelsystem parametrierten, material- und werkzeugspezifischen Grenzwerten [FANU04a, PROM06] oder mit für den Bearbeitungsfall eingelernten Referenz- [FRAN06] und Grenzwertkurven [ARTI03]. Das Ziel der ACC Systeme ist es, den Vorschub so zu regeln, dass das Potential der Werkzeuge in Abhängigkeit von den Eingriffsbedingungen und dem Verschleißzustand ausgenutzt wird. Anwendern zufolge führt die mögliche Steigerung der Vorschubgeschwindigkeit zu einer Verkürzung der Bearbeitungszeit von bis zu 40% [DOWE03]. Vorhersehbare Unregelmäßigkeiten blenden die Systeme getriggert durch spezifische Befehle im NC-Programm oder zu diskreten Zeiten relativ zur Programmabarbeitung aus [ARTI03]. Beispiele hierfür sind An- und Abfahrbewegungen oder Schnittunterbrechungen. Nachteilig an diesen Verfahren ist die fehlende direkte Kopplung mit dem NC-Programm. Insbesondere bei eingelernten Grenzwertkurven muss der Lernprozess nach Programmänderungen an der Abarbeitungsfolge, der Bahnführung, den Technologievorgaben oder dem Werkzeug wiederholt werden

Ein weiteres Beispiel für eine adaptive Vorschubregelung findet sich in der EDM Bearbeitung (Electro-Discharge-Machining / Erodieren). Die Vorschubregelung hat die Aufgabe, den Arbeitsspalt zwischen Werkstück und Elektrode so einzustellen, dass einerseits kein Kurzschluss durch direkten Kontakt entsteht und andererseits der Spalt nicht zu groß wird, so dass keine prozesstypischen Entladungen mehr stattfinden können. Dazu wird der Spannungsverlauf als Prozesskenngröße überwacht und die Spaltweite über den Vorschub geregelt. Der Verfahrweg des Werkzeugs ist jedoch fest durch das NC-Programm vorgeben und wird nicht manipuliert.

Adaptive Bahnführung

Beim Laserschweißen muss der Fokus des Laserstrahls mit hoher Genauigkeit zum Fügespalt positioniert werden. Jedoch weicht die Lage des Spalts aufgrund ungenauer Nahtflanken, Spannfehler und des prozessbedingten Wärmeverzugs von der geplanten und im NC-Programm kodierten Nahtführung ab. Die Industrie setzt daher Nahtfolgesensoren ein, um den Verlauf des Fügespalts zu detektieren. Zur Lagekorrektur des Fokus wird die ermittelte Abweichung mit den Sollwerten der Maschinensteuerung verrechnet. Je nach Steuerung findet diese Verrechnung in der Vorlaufsteuerung (z.B. ABB) oder in einer externen Sollwertmanipulation (z.B. Kuka) statt [META06]. Es handelt sich dabei um eine reine Positionskorrektur innerhalb der Bahnführung, d.h. eine Verrechnung von Offsets, und nicht um eine regelbare Bahnplanung.

2.3.2. Entwicklungsansätze in der Prozessüberwachung und –regelung

Developments in process monitoring and control

Neben den industriell eingesetzten Systemen gibt es eine Vielzahl von Forschungs- und Entwicklungskonzepten. Einzelne dieser Lösungen werden im Folgenden näher betrachtet und dabei ihre Potentiale und Einschränkungen herausgearbeitet.

Modellbasierte Werkzeugüberwachung

Marktübliche Werkzeugüberwachungssysteme arbeiten mit bearbeitungsspezifisch eingelernten Überwachungsmodellen oder parametrierten Grenzwerten. Damit verbunden ist ein erhöhter Aufwand, wenn wechselnde Rohteilabmaße und Fertigungsgeometrien bearbeitet werden müssen. Mit dem Ziel, die Überwachung an die Bearbeitungsgeometrie zu koppeln, wurde von *Rehse* im Rahmen des Sonderforschungsbereiches „Autonome Produktionszellen“ (SFB 368) eine modellbasierte Werkzeugüberwachung für das Bohren und Messerkopfstirnfräsen entwickelt. Ausgehend von dem gewählten Werkzeug, Material und dem zu zerspanenden Querschnitt, berechnet er die zu erwartenden Bearbeitungskräfte. Dabei werden die Rohteilgeometrie als auch der Bearbeitungsfortschritt berücksichtigt, so dass Schnittunterbrechungen und wechselnde Eingriffsverhältnisse an Querbohrungen oder Absätzen positionsgenau in die Berechnung eingehen. Das Einlernen einer Schnittkraftcharakteristik entfällt. Der reale, gemessene Prozess wird direkt mit dem theoretischen Schnittkraftmodell verglichen [REHS98]. Die bahn- und schneidenbezogene Berechnung des Zerspanungsquerschnitts und der resultierenden Bearbeitungskräfte setzt jedoch eine hohe Rechenleistung, umfangreiche Eingangsinformationen über Geometrie und Werkzeug sowie eine enge Kopplung mit der Programmverarbeitung und der Lageregelung zur Synchronisation des Modells voraus. Offline gerechnete Modelle haben zudem den Nachteil, dass sie für die einzelnen Schneiden definierte Winkellagen voraussetzen müssen. Diese ergeben sich jedoch im realen Prozess zufällig, da Fräswerkzeuge über die Drehzahl und nicht den Winkel geregelt werden. Insbesondere bei unterbrochenen Schnitten oder variierenden Umschlingungswinkeln kommt es so zu Abweichungen.

Kompensation von Werkzeugverformungen

Abdrängungen des Werkzeugs aufgrund von Bearbeitungskräften führen zu Maß- und Formabweichungen am Werkstück. Neben Ansätzen, die Abdrängung bei der NC-Programmierung vorzuhalten [TOEN01], gibt es Systeme, die diese Fehler im realen Bearbeitungsprozess bestimmen und durch eine Korrektur der Werkzeugstellung kompensieren. Hierzu werden die Bearbeitungskräfte über eine Kraftmesssensorik oder die Ströme der Achsantriebe und der Hauptspindel ermittelt. Aus den Kräften lässt sich die Verformung des Werkzeugs in Richtung der Maschinenachsen errechnen. Dieser Wert wird als Offset-Signal an den Lageregler übermittelt, um die Sollposition um den Anteil der Abdrängung zu korrigieren [WECK01d]. Diese Ansätze können die negativen Effekte einer Werkzeugverformung und -abdrängung nur kompensieren. Zur Vermeidung müssten die Prozessparameter entsprechend der Steifigkeit von Werkstück, Werkzeug und Maschine abgestimmt werden. Ein Beispiel wären moderatere Zerspanungswerte und eine Anpassung der Schnittaufteilung.

Regelsysteme zur Rattervermeidung

Bei der Beurteilung des dynamischen Verhaltens von Werkzeugmaschinen wird zwischen fremd- oder selbsterregten Schwingungen unterschieden. Fremderregte Schwingungen sind auf externe Störungen oder Unwuchten zurückführen. Sie lassen sich nicht über die im NC-Programm kodierten Prozessparameter oder die Bearbeitungsstrategie abstellen. Selbsterregte Schwingungen entstehen dagegen erst im Prozess. Sie hängen von den gewählten Bearbeitungsparametern und dem dynamischen Nachgiebigkeitsverhalten des Maschine-Werkzeug-Werkstück Systems ab. Die Auslöser sind oftmals Störungen oder periodisch wechselnde Prozesskräfte. Diese führen zu elastischen Verformungen und damit zu einer Relativverlagerung zwischen Werkzeug und Werkstück. Die Folge ist eine Änderung der Spandicke, so dass der nächste Schnitt (Drehen), bzw. die nachfolgende Schneide (Fräsen) einen größeren Querschnitt zerspanen muss. Die zwischen Werkstück und Werkzeug wirkenden Kräfte steigen an und aufgrund der Nachgiebigkeit des Systems kommt es erneut zu einer Relativverlagerung. Wiederholt sich der Effekt mit einer der Eigenfrequenzen des Systems, spricht man von regenerativem Rattern. Neben den Oberflächenfehlern können die resultierenden Kräfte zu Schäden am Werkstück, Werkzeug und der Maschine führen [WECK77]. Im Prozess lässt sich das

Rattern über die Spanungsparameter regeln. In Bild 2.12 ist für einen Fräsprozess die sogenannte Stabilitätskarte dargestellt, die in Abhängigkeit von Material, Werkzeug, Maschine und den Parametern Schneidenzahl, Drehzahl und Schnittbreite stabile und aufgrund von Rattern instabile Bereiche ausweist.

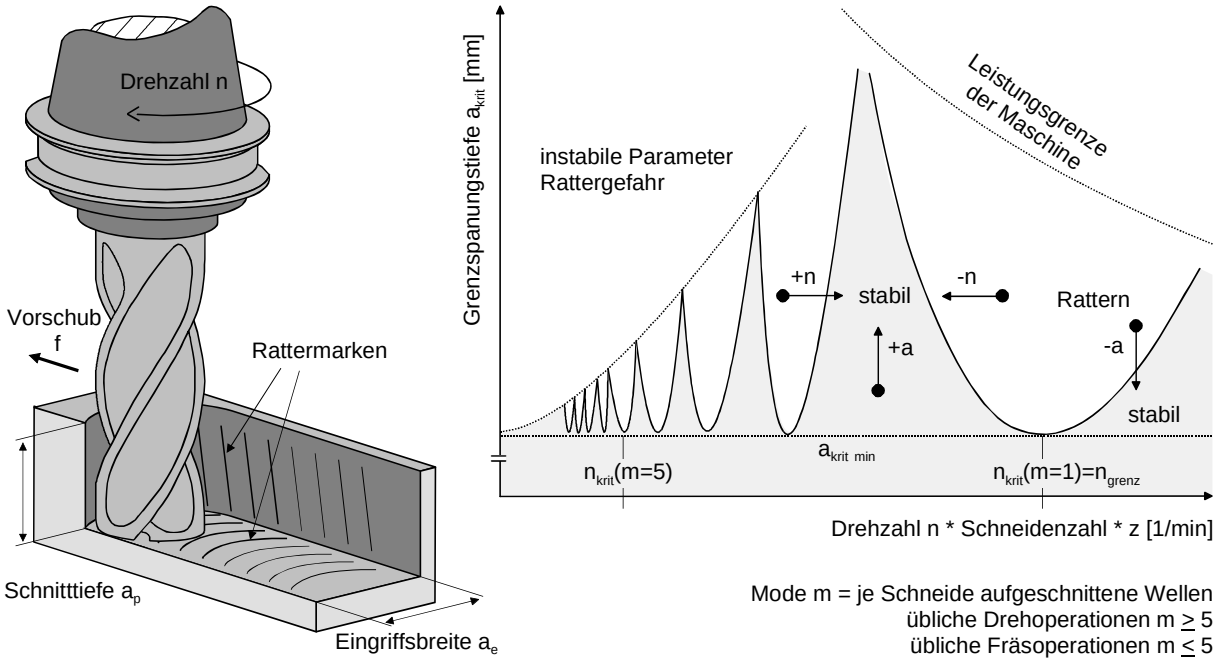


Bild 2.12: Rattermarken und Stabilitätskarte für einen Fräsprozess
Chatter marks and stability chart for a milling process

Adaptive Drehzahlregelung

Ausgehend von den Stabilitätskarten schlägt *Tlusty* als Regelparameter gegen das regenerative Rattern die Spindeldrehzahl n vor. Über einen Körperschallsensor detektiert er das Rattern und variiert die Drehzahl, um den Prozess zu stabilisieren [TLUS92]. In der Industrie haben sich derartige Systeme für die Dreh-, Fräs- und Bohrbearbeitung bislang nicht durchgesetzt. Mögliche Gründe lassen sich auf den technischen Aufwand und die Reaktionszeit der Systeme zurückführen. Ferner sind bis zum Erkennen der Störung in der Regel bereits Rattermarken auf der Bauteiloberfläche entstanden. Zudem würde sich mit der Spindeldrehzahl bei gleich bleibendem Vorschub die kinematische Rauhtiefe ändern. Die Folge wären unerwünschte Oberflächenunregelmäßigkeiten.

Adaptive Schnitttiefenregelung

Eine weitere effektive Regelstrategie gegen das regenerative Rattern ist die Reduktion der Zerspanungstiefe. Daraus resultiert für die Dreh- und Fräsbearbeitung eine prozessabhängige Schnittaufteilung zur Laufzeit. Für die Drehbearbeitung gibt es Lösungskonzepte, die auf einer prozessabhängigen Regelung der Zustellung beruhen. Ausgehend von der adaptierten Schnitttiefe berechnet die Steuerung zusätzliche, zur ursprünglichen Kontur asymptotisch verschobene Bahnen und fügt diese in den Bearbeitungsablauf ein. Dieser Vorgang wird solange wiederholt, bis die programmierte Endkontur erreicht worden ist [GIES73].

Auf die Fräsbearbeitung lässt sich dieses 2-dimensionale Konzept der Drehbearbeitung nicht direkt übertragen. Hierzu müsste die Programmverarbeitung der NC-Steuerung eine Schnittaufteilung unter Berücksichtigung von Hinterschnitten und Kollisionen durchführen. Das ist mit kommerziellen NC-Steuerungen nicht möglich [WECK01c]. Für die Fräsbearbeitung schlägt *Versch* daher einen vereinfachten, auf die Adaption der Schnitttiefe begrenzten Ansatz

vor. Er kodiert die axiale Schnitttiefe mit NC-Variablen. In einem rampenförmigen Lernschnitt bestimmt er dann die ideale Schnitttiefe, speichert sie in den Variablen und übernimmt sie so in das auszuführende Bearbeitungsprogramm. In einer Schleife werden die programmierten Bahnsegmente solange wiederholt, bis durch eine schrittweise Zustellung um die ermittelte Schnitttiefe die Sollvorgabe erreicht ist [VERS04]. Dieser Ansatz hat verschiedene Nachteile. Zum einen handelt es sich anstelle einer Regelung um eine Steuerung, da der aktuelle Prozessverlauf nicht berücksichtigt wird. Zudem wird die Programmierung komplizierter, da die alternativen Bewegungsbahnen vorgeplant und parametriert werden müssen. Als Steuerparameter steht nur die axiale Schnitttiefe, nicht jedoch die Eingriffsbreite oder die Strategie der Schnittaufteilung zur Verfügung.

Adaptive Schnittaufteilung

Eine komplexere, mehrdimensionale Lösung wurde von *Brouër* für die Fräsbearbeitung von ebenen Taschenproblemen vorgestellt. Auf der Grundlage einer im Rahmen des SFB 368 entwickelten featureorientierten NC-Programmierschnittstelle „STEP-NC“ berechnet er die Schnittaufteilung zur Laufzeit in der NC-Steuerung. Sein Konzept sieht vor, dass bei Prozessstörungen, wie dem regenerativen Rattern, neue Verfahrwege mit reduzierter Schnitttiefe und -breite berechnet werden [BROU00]. Da die Bahnauflösung jedoch nicht mit der Ausführung im NC-Kern synchronisiert ist, kann die Korrektur nur an den Programmstellen erfolgen, zu denen das Werkzeug aus der Bearbeitung zurückgezogen wurde. Als Beispiel führt *Brouër* die Adaption einer nachfolgenden Zerspanebene an, deren Bahnsegmente innerhalb der Programmverarbeitung als eine Liste sogenannter Sub-Workingsteps zusammengefasst ist und als solches geschlossen verworfen und durch eine neu generierte Liste ersetzt wird. Den tangentialen Übergang von einem Sub-Workingstep in einen neuen, adaptieren Workingstep leistet der Ansatz nicht, da die Workingsteps nicht interagieren, um Parameter oder Positionen anzugleichen. Die Eingriffsmöglichkeiten sind während der Bearbeitung damit auf die Manipulation der Override und maschinenseitige Aktuatoren begrenzt, Bild 2.13.

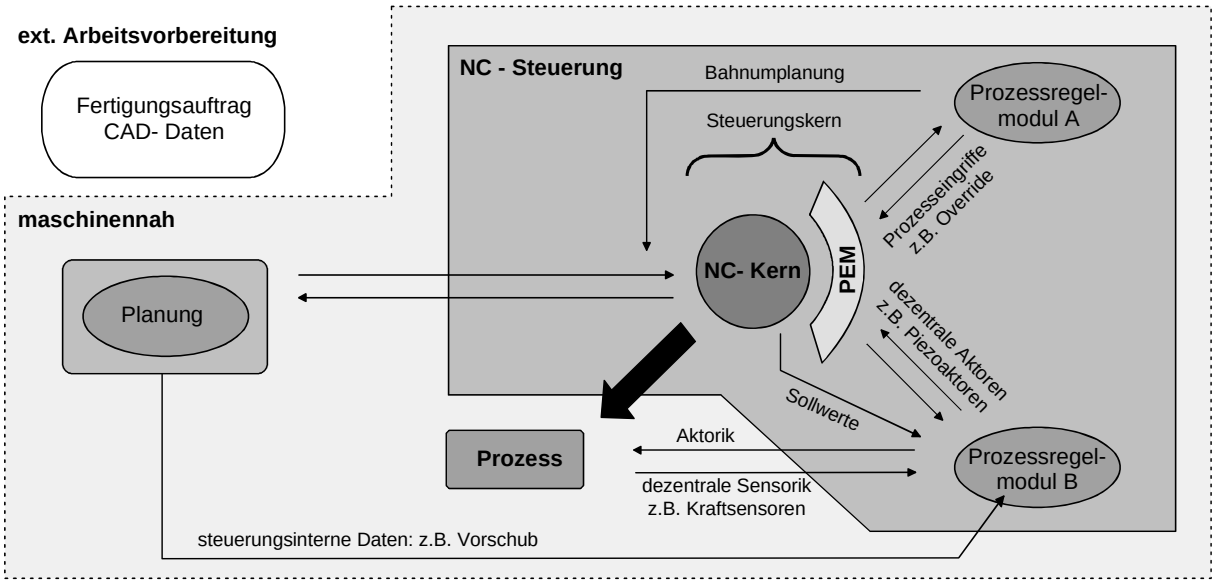


Bild 2.13: Prozesskorrektur durch Bahnauflösung in der Satzvorbereitung (nach [BROU00])
Process manipulation by adapting the toolpath generation in the pre-processing

2.3.3. Steuerungsintegration von Überwachungs- und Regelungssystemen

Integration of monitoring and control systems into the numerical control

Wie bereits dargestellt, haben offene Kommunikations- und Entwicklungsschnittstellen zur NC-Steuerung sowie die digitale Antriebstechnik die Realisierung von Überwachungs- und Regelungssystemen in den letzten Jahren vereinfacht. Darüber hinaus erlauben sie die direkte Integration der Systeme in die NC-Steuerung und den Verzicht auf externe Sensorik. Die Prozessdaten lassen sich aus den digitalen Antriebsdaten auf der Basis der Leistungsaufnahme und dem damit korrelierenden Antriebsmoment rekonstruieren [SERC06a]. Die Integrationskonzepte der verschiedenen Systeme reichen von externer Rechner- und Sensorhardware [NORD04] über Einsteckkarten [ARTI00], bis hin zu vollständig softwarebasierten, steuerungsintegrierten Lösungen, ohne zusätzliche externe Sensorik [SCHL03]. Die Vorteile der Systemintegration liegen in dem schnelleren, direkten Zugriff auf steuerungsinterne Werkzeug-, Maschinen- und NC-Programmdaten, die SPS- und Antriebssignale sowie die zu regelnden Sollgrößen. Ferner lässt sich die Benutzerschnittstelle der NC-Steuerung zur Dateneingabe und zur Visualisierung nutzen. Darüber hinaus ist eine funktionale Kopplung der Systeme mit der Steuerung wichtig, um die aus der Überwachung resultierenden Aktionen auszuführen und die prozessbestimmenden Bearbeitungsparameter zu regeln.

Externe Systeme

Eine Vielzahl der industriell eingesetzten Lösungen basiert auf eigener Sensorik und Hardware zur Prozessüberwachung und -auswertung. Sie haben eine eigene Benutzerschnittstelle und sind nicht in die Oberfläche der NC-Steuerung integriert. In die laufende Bearbeitung greifen sie über SPS Befehle oder eine Manipulation der Sollwerte ein. Bis auf die Möglichkeit NC-Variablen zu setzen, nehmen die externen Systeme keinen Einfluss auf das NC-Programm und dessen steuerungsinterne Abarbeitung. Die Synchronisation mit dem Bearbeitungsablauf ist auf den Abruf von SPS Signalen (Programmstart, Werkzeugwechsel, …) und den Zugriff auf Sollwerte (Vorschub, Drehzahl, …) begrenzt.

Für eine steuerungsübergreifend einheitliche Anbindung an die SPS haben die Firmen Brankamp, Artis und Prometec das Brankamp-Artis-Prometec-Standard-Interface (BAPSI) spezifiziert. Darin sind 24 von der SPS bereitzustellende Signale festgelegt. Unter anderem gehören hierzu das An- und Abschalten der Überwachung, der Code des aktiven Werkzeugs, die ID des NC-Programms und die Art des Verfahrbefehls (Eilgang, definierter Vorschub). Über diese Informationen wird die Überwachung parametriert und gestartet. Die Reaktionen des Überwachungssystems werden über 8 Meldesignale an die SPS übermittelt. Zu den Meldungen zählen Stopp-Befehle, Quittieranforderungen, Warnungen und Befehle an die Werkzeugverwaltung [BAPS97]. Nicht enthalten sind Zugriffe auf das NC-Programm oder die Sensorsignale der Antriebe.

Integrierte, auf zusätzlicher Hardware basierende Systeme

Mit dem Ziel, die Funktionen und Daten der NC-Steuerung nutzen zu können und gleichzeitig eigene Sensorik einzubinden sowie die Rechen- und Speicherleistung an die individuellen Bedürfnisse der Überwachung und Regelung anzupassen, setzen Drittanbieter Einsteckkarten mit eigenem Prozessor ein. Die Überwachungs- und Regelungsfunktionen werden auf der Einsteckkarte appliziert und durch Erweiterungen der Steuerungssoftware (z.B. Siemens NCK Compile-Zyklen) mit der Steuerung gekoppelt. Über den Steuerungsbus sowie die Backplane, bzw. das Motherboard des Steuerungsrechners werden die Daten und Befehle ausgetauscht. Die Anzeige und Bedienung erfolgt über systemeigene Applikationen in der Benutzeroberfläche der Steuerung.

Der Funktionsumfang des Überwachungs- und Regelsystems kann über die Applikationen auf der Zusatzhardware als auch über die des NC-Kerns bedarfsgerecht erweitert werden. Bei-

spielsweise bieten Systemhersteller Funktionen zur Werkzeugverschleißüberwachung, zum Bearbeitungsabbruch bei Werkzeugbruch und Kollision sowie zur verschleißabhängigen Vorschubmaximierung an [ARTI03]. Zudem können die Systeme prozessabhängig gesteuert werden. Dazu werten sie die steuerungsseitigen Weginformationen aus, erkennen Eilgangbewegungen („G 0“) und blenden diese aus der Überwachung aus [ARTI02].

Die Systeme nutzen die steuerungsseitig verfügbaren SPS Signale und Sollwerte, um Aktionen auszulösen oder den Prozess zu regeln. Dagegen finden sich am Markt keine Lösungen, mit denen sich die NC-Programmvorgaben, d.h. der programmierte Prozess, und dessen Umsetzung zur Laufzeit regeln lassen.

Integrierte, softwarebasierte Systeme

Einzelne Steuerungshersteller bieten vollständig integrierte, nur auf der Steuerungshardware basierende Softwaresysteme zur Überwachung an. Ein Beispiel ist die Bohrerbruchüberwachung in der *MTC* Steuerung der Bosch Rexroth. Dabei werden in der NC-Steuerung die digitalen Daten der SERCOS Antriebe ausgewertet, überwacht und der Prozess beim Überschreiten von Grenzwerten abgebrochen. Der Anwender steuert die An- und Abwahl der Überwachungsfunktionen über spezielle Befehle im NC-Programm [INDR98]. Eine prozessabhängige, automatisierte Überwachung auf der Basis von Bohrzyklen, dem Vorschub oder der Art des Werkzeugeingriffs ist nicht vorgesehen. In die Prozessführung greifen die industriellen Systeme nur auf der Sollwertebene regelnd ein [FANU04a]. Prozessbestimmende Parameter, wie beispielsweise die An- und Abfahrstrategien, die Schnittaufteilung oder die Werkzeugwahl, werden nach Programm abgearbeitet.

2.4. Fazit

Resume

Sowohl bei der manuellen NC-Programmierung als auch bei den verschiedenen rechnerunterstützten Programmierverfahren ist es das vorrangige Ziel, prozesssichere Bearbeitungsprogramme zu generieren. Dabei ist allen Verfahren gemein, dass sie im Vorfeld der Bearbeitung eingesetzt werden und daher von einem angenommenen, idealisierten Prozessmodell ausgehen müssen, Bild 2.14 oben. Abweichungen und Streuungen des realen Prozesses werden in der Planung durch Sicherheitsbeiwerte abgeschätzt. Die Folge sind suboptimale, moderate Technologieparameter und Bearbeitungsbahnen. Im Fall von Fehlannahmen kann es zudem zur Überlastung kommen und damit zu negativen Auswirkungen auf die Standzeit der Werkzeuge, die Belastung der Maschine oder das Bearbeitungsergebnis.

Ein weiterer Nachteil der NC-Verfahrenskette resultiert aus den begrenzten Möglichkeiten zur Programmanpassung an der Maschine. Die Ursache hierfür liegt in dem auf Schalt- und Wegbefehle reduzierten Informationsgehalt des NC-Programms, Bild 2.14 Mitte. Es fehlt die Grundlage für eine Umplanung des Ablaufs oder die Neuberechnung der Technologieparameter und der daraus resultierenden Schnittaufteilung. Während des Einfahrens und Optimierens geht so wertvolle Zeit für Iterationsschleifen mit der Bearbeitungsplanung und NC-Programmierung verloren.

Darüber hinaus können Fehler bei der NC-Programmierung, Abweichungen der Bearbeitungsaufmaße oder Störungen im Bearbeitungsprozess nicht sicher ausgeschlossen werden. Um direkt an der Maschine in den Prozess eingreifen zu können, gibt es daher Handschalter in der Benutzerschnittstelle der Steuerung sowie automatisierte Überwachungs- und Regelungssysteme. Diese Eingriffsmöglichkeiten sind jedoch von den Schnittstellen und Funktionen der NC-Steuerung abhängig, wodurch sie in der Regel auf eine Manipulation der Sollwerte für Drehzahl, Vorschub oder die Achspositionen beschränkt sind, Bild 2.14 unten.

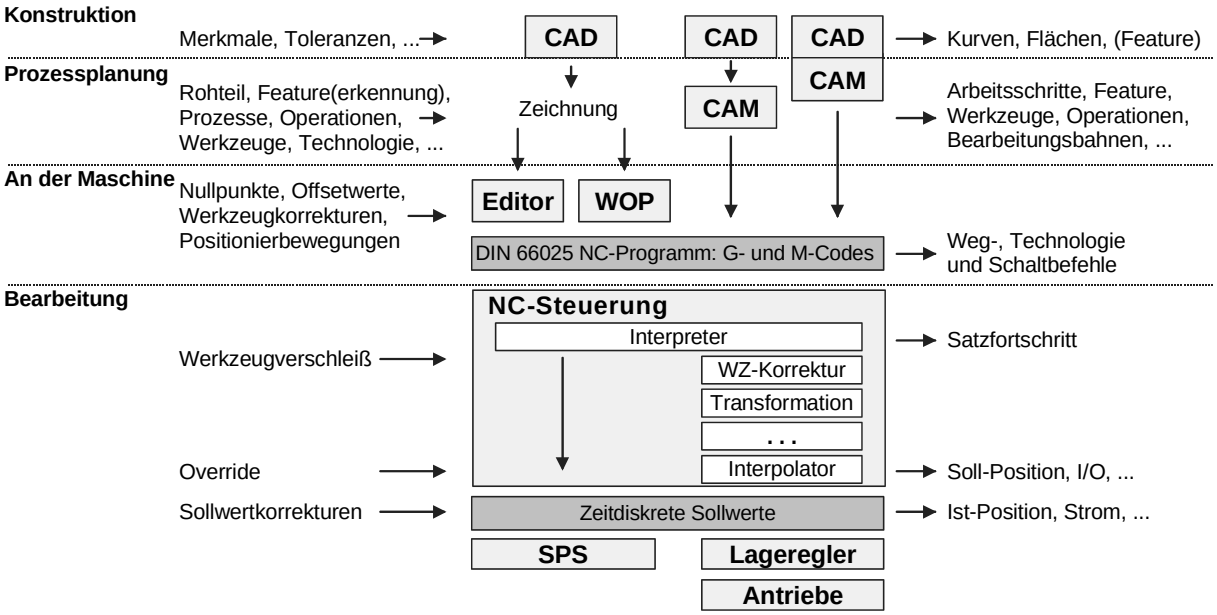


Bild 2.14: Heutiger Informations- und Funktionsverlust von der Planung zur Bearbeitung
Today's loss of information and functionality from planning to machining

Eine Lösung, um das NC-Programm und dessen Abarbeitung während der Bearbeitung zu manipulieren, ist industriell nicht verfügbar. Hierzu fehlt es zum einen an der notwendigen Informationsbereitstellung über eine von der Planung bis in die NC-Steuerung durchgängige, aufgabengerechten NC-Programmierschnittstelle. So tauschen CAM-Systeme ihre Datensätze untereinander praktisch gar nicht aus und zur NC-Steuerung werden nur Weg- und Schaltbefehle oder Zyklen übertragen. Zum anderen fehlen in der NC-Steuerung die notwendigen Algorithmen, mit denen es möglich wäre, die Aufgaben-, Operations-, Technologie- und Bahnvorgaben eines höherwertigen NC-Programms an der Steuerung anzuzeigen, zu editieren und während der Bearbeitung zu verfolgen und zu manipulieren.

3. Zielsetzung und Vorgehensweise

Objective and approach

Das vorrangige Ziel der numerisch gesteuerten Fertigung sind optimierte, störungsfreie Prozesse, um die Produktionsanlagen maximal auszulasten und qualitativ hochwertige Produkte wettbewerbsfähig zu fertigen. Hierzu werden in den Bereichen der NC-Programmierung, der NC-Steuerung sowie der Prozessüberwachung und -regelung fortlaufend neue Funktionen und Systeme entwickelt. Die Betrachtung des Stands der Technik zeigt, dass diese Entwicklungen zum einen durch eine unzureichende Aufgabenbeschreibung im NC-Programm und zum anderen durch fehlende Steuerungsfunktionen zur bearbeitungsparallelen Manipulation des NC-Programminhalts und dessen steuerungsinterner Ausführung eingeschränkt werden.

Insbesondere fehlen beim Einfahren neuer und beim Korrigieren kritischer Prozesse facharbeitergerechte, prozessorientierte Hilfen. So lassen sich in der Regel nur Steuerungsparameter, wie der Bahnvorschub, die Spindeldrehzahl oder die Wirkleistung während der laufenden Bearbeitung manipulieren. Andere prozessbestimmende Parameter, wie die Zustellung, die Bahnführung oder das Werkzeug lassen sich dagegen nur nach Abbruch der Bearbeitung im NC-Programm neu kodieren. Dabei geht wertvolle Produktionszeit verloren. Zudem kann das Frei- und Wiederanfahren des Werkzeugs zu Fehlern am Werkstück oder Kollisionen führen.

3.1. Zielsetzung

Objective

Vor dem Hintergrund dieser Einschränkungen ist es das Ziel der vorliegenden Arbeit, ein System zu entwickeln, mit dem sich der Bearbeitungsprozess während der Abarbeitung des Programms durch Eingriffe in die NC-Programmvorgaben und deren steuerungstechnische Abarbeitung in der NC-Steuerung beeinflussen lässt. Dadurch sollen einerseits die programmierten Technologieparameter, Werkzeuge und Strategien als auch die Bahnauflösung zur Prozesskorrektur genutzt werden. Ferner gilt es, zeitintensive Unterbrechungen zu vermeiden, wie sie heute zur Korrektur und Optimierung des NC-Programms notwendig sind.

Die erforderlichen Informationen sollen nicht mehr als zusammenhanglose Schalt- und Wegbefehle nach DIN 66025 kodiert werden. Stattdessen wird eine neue Programmierschnittstelle definiert, die es ermöglicht, von der Planung bis in die Bearbeitung Prozesse durchgängig in Form von Arbeitsschritten, Featuren und Operationen zu programmieren, Bild 3.1 Mitte.

Bezogen auf den Facharbeiter an der NC-Maschine soll dieses System das Einfahren neuer Programme unterstützen und korrigierende Eingriffe in laufende, suboptimale oder instabile Prozesse ermöglichen. Dazu müssen ihm die Bearbeitungsaufgabe, die Prozessvorgaben, die anstehende Programmumsetzung in der Steuerung als auch der aktuelle Verlauf des Bearbeitungsprozesses dargestellt werden. Bei der Interaktion mit dem System soll er in Abhängigkeit von dem eigenem Prozessverständnis und seinen Befugnissen die Möglichkeit haben, Parameter auf der Basis des NC-Programms oder in Form vordefinierter Steuerungsstrategien zu ändern. Hierfür sind sowohl in der NC-Programmierschnittstelle, der Benutzungsoberfläche als auch im Kern der NC-Steuerung Schnittstellen und Funktionen zu entwickeln.

Ein weiteres Ziel ist es, Optimierungen und Korrekturen im laufenden Prozess zu erfassen und für eine Folgebearbeitung vorzuhalten. Dazu muss der Werker die Änderungen und deren Auswirkung parallel zur laufenden Bearbeitung, als auch im Nachgang erkennen und nachvollziehen können. Die Ergebnisse eines Optimierungsprozesses sollen sich nach Möglichkeit direkt in das NC-Programm übertragen und in nachfolgenden Bearbeitungen nutzen lassen.

Den Entwicklern von Algorithmen zur NC-Programmumsetzung, Prozessüberwachungen und -regelungen werden diese flexiblen Systemfunktionen als offenes, erweiterbares Applikationsprotokoll zur Verfügung gestellt. Sie bieten eine Plattform an, in die sich anwenderspezifische, spezialisierte Lösungen mit geringem Aufwand integrieren lassen. Den individuellen Lösungen der Entwickler soll der Zugriff auf die NC-Programminformationen, deren steuerungsinterne Umsetzung sowie die Prozessverfolgung offen gelegt werden, um die Bearbeitung und eine Prozessüberwachung optimal an die aktuelle Situation anpassen zu können. Dem übergeordnet muss das Grundsystem der NC-Steuerung sicherstellen, dass die Funktionen sich nicht gegenseitig blockieren und die Maschine zu jedem Zeitpunkt der Bearbeitung in einen sicheren Zustand überführt werden kann.

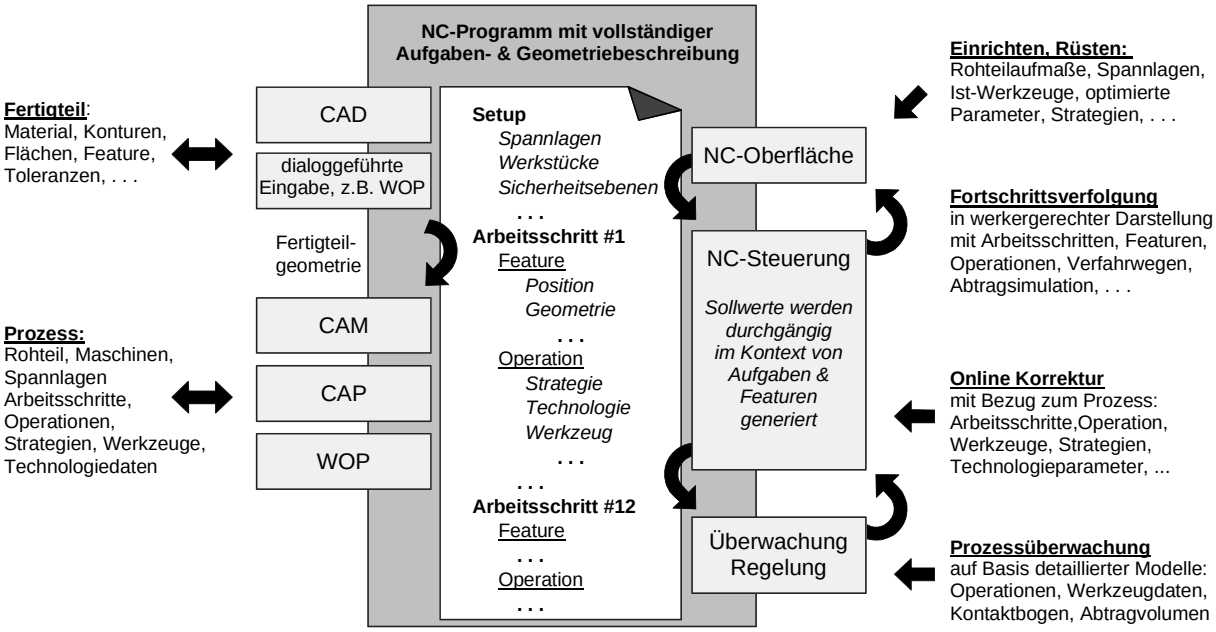


Bild 3.1: Durchgängig verfügbare Aufgaben- und Geometriebeschreibung von der NC-Programmierung über die Abarbeitung bis in die Überwachung und Regelung

Continuously accessible task and geometry description from NC-programming to program execution in the numerical control, process monitoring and control

3.2. Vorgehensweise

Approach

Im Stand der Technik wurde gezeigt, dass entlang der NC-Prozesskette abgeschlossene Bereiche durchlaufen werden, die durch spezialisierte Computersysteme unterstützt werden. Diese grenzen sich voneinander durch definierte Schnittstellen ab, die jedoch den Informationsgehalt schrittweise auf die minimalen, zur Ausführung notwendigen NC-Programmcodes reduzieren. Für aufgaben- und prozessorientierte Eingriffe an der Maschine fehlt folglich die notwendige vollständige Informationsgrundlage. Daher werden ausgehend von einer Anforderungsanalyse steuerungstechnische Lösungen erarbeitet, die die erforderlichen Informationen im NC-Programm bereitstellen, um den Prozess aufgaben- und operationsorientiert verfolgen und in der laufenden Bearbeitung umfangreich manipulieren zu können.

Übergreifend werden in Kapitel 0 die Anforderungen an eine steuerungsintegrierte prozessabhängige Programmbeeinflussung und die damit verbundene Bahnauflösung herausgearbeitet. Hierzu werden zunächst die technologisch relevanten Parameter und Programmelemente

definiert, die das System als Stellgrößen bereitstellen soll. Zudem sind die Anforderungen der Systementwickler und Anwender zu analysieren.

Aufbauend auf den erarbeitenden Anforderungen wird in Kapitel 5 ein allgemeingültiges Konzept für das Gesamtsystem einer steuerungsintegrierten, prozessabhängigen NC-Programmverarbeitung entwickelt. Dabei findet insbesondere die Anforderung Beachtung, hersteller- und anwendungsspezifische Funktionen in einem aufgaben- und prozessorientierten NC-Programm kodieren und deren Abarbeitung in die NC-Steuerung integrieren zu können. Des Weiteren bezieht das Konzept bestehende Abläufe und Strukturen ein und zeigt Migrationsschritte auf, um konventionelle Steuerungen schrittweise im Sinne der Aufgabenstellung weiterzuentwickeln.

In Kapitel 6 und 7 werden der Aufbau und die Funktionsweise der Teilsysteme NC-Steuerungskern und Benutzerschnittstelle detailliert. Im Fokus stehen dabei die Programmverarbeitung und die Kopplung der beiden Systeme, so dass in der laufenden Bearbeitung Änderungen an Vorgaben des NC-Programms direkt ausgewertet und umgesetzt werden können.

Die Realisierung des entworfenen Systems wird in Kapitel 8 am Beispiel der Fräsbearbeitung prismatischer Merkmale erläutert. Der hierzu entwickelte Demonstrator besteht aus einem System zur featureorientierten NC-Programmierung, einer NC-Steuerung mit aufgabenorientierter Programmverarbeitung und Schnittstellen zur Prozessrückkopplung sowie eingebundenen Modulen zur Prozessüberwachung und 3D Prozessverfolgung durch den Werker. Das Kapitel schließt mit Zeitmessungen aus praktischen Bearbeitungsversuchen. Diese demonstrieren die Leistungsfähigkeit der entwickelten Bahnauflösung sowie deren bearbeitungsparallele Adaption.

Kapitel 9 beschließt die Arbeit mit einer Zusammenfassung und dem Ausblick auf mögliche Formen der industriellen Umsetzung sowie der dazu zu leistenden Entwicklungsarbeit.

4. Anforderungen an die Programmierung und Programmverarbeitung

Requirements on NC-programming and NC-processing

Im Rahmen dieser Arbeit wird ein NC-Steuerungssystem entwickelt, das über den Zugriff auf höherwertige NC-Programminformationen und deren Verarbeitung qualitativ und quantitativ erweiterte Eingriffsmöglichkeiten in den Bearbeitungsablauf ermöglicht. Dazu sind Steuerungsfunktionen zu entwickeln, mit denen diese Informationen manipuliert und ihre Umsetzung synchron zur laufenden Bearbeitung adaptiert werden können. Der Werker als auch eine automatisierte Prozessregelung sollen neben den bekannten Regelmechanismen für Vorschub, Drehzahl oder Leistung die programmierten Operations- und Technologieparameter nutzen können, um die programmierte Bearbeitung einzufahren, zu optimieren oder auf Störungen zu reagieren.

Im Folgenden werden zunächst die Randbedingungen und die Anforderungen einer NC-programmbasierten Prozessregelung analysiert. Dabei werden drei Teilbereiche unterschieden:

- Die Informationsbereitstellung im NC-Programm (Kap. 4.1)
- Die adaptive Programmverarbeitung in der NC-Steuerung (Kap. 4.2)
- Die Benutzerschnittstelle zur Programmierung und Prozessmanipulation (Kap. 4.3)

4.1. Anforderungen an das NC-Programm

Requirements on the NC-program

In einem NC-Programm werden der steuerungstechnische Ablauf und die prozessbestimmenden Technologieparameter der Bearbeitung kodiert und über die NC-Steuerung zur Ausführung gebracht. Studien der NC- und der Fraunhofer Gesellschaft belegen, dass der Werkstatt und dem Werker zentrale Rollen bei der Erstellung, Erprobung und Optimierung von NC-Programmen zukommen [NCG93, LAYG05]. Mit dem Ziel, qualitativ und wirtschaftlich optimal zu fertigen, fordert die Industrie daher NC-Programmierschnittstellen und Systeme, die den Werker besser bei der Programmerstellung und der Programmoptimierung an der Maschine unterstützen. Die Ursache für Defizite wird in der abstrakten, befehls- und steuerungsorientierten Programmkodierung nach DIN 66025 gesehen. Darüber hinaus verfolgt die Industrie das Ziel, weitreichendere Probleme der NC-Programmierung in den Bereichen des Datenmanagements, der Arbeitsvorbereitung und der Programmabarbeitung zu lösen [GEHR89]. Diese Forderungen haben Einfluss auf zukünftige NC-Programmierschnittstellen und folglich auf das zu entwickelnde Gesamtkonzept einer Prozessregelung auf der Basis der NC-Programmvorgaben.

Datenmanagement: Die unterschiedlichen Datenformate der zur NC-Programmierung eingesetzten Systeme führen zu Konvertierungsverlusten, zu Fehlern und, da Änderungen manuell in den Teilsystemen nachgeführt werden müssen, zu Inkonsistenzen. Ein durchgängiges Datenformat von der Konstruktion über die Arbeitsvorbereitung bis in die NC-Steuerung würde diese Defizite ausschließen und Unternehmen darüber hinaus die Möglichkeit eröffnen, das zentrale Produktmodell um ein Prozessmodell mit den zur Bearbeitung erforderlichen Ressourcen und technologischen Vorgaben zu erweitern (Vergleichbar mit dem STEP Standard und seinen Applikationsprotokollen ISO10303): aus Konstruktionsmerkmalen werden Bearbeitungsmerkmale, d.h. fertigungsgerechte Feature, die sich direkt einer Bearbeitungstechnologie zuordnen lassen [VANJ98, HAAS00] und die von der NC-Steuerung in Werkzeugwege aufgelöst werden können [BROU00].

Ferner möchte die Industrie auf die teuren und pflegeintensiven Postprozessoren verzichten und zwischen den verfügbaren Maschinen die NC-Programme flexibel austauschen können [WEYR00]. Anstelle einer steuerungsspezifischen Befehlsfolge muss das NC-Programm eine umfassende, steuerungsneutrale Aufgabenbeschreibung enthalten, die sich durch den Werker an der Maschine ändern und von der Steuerung maschinenorientiert verarbeiten lässt [STEP99].

Arbeitsvorbereitung: Der Prozess- und Operationsplanung stehen heute weder leistungsfähige Standards zur Beschreibung von Bearbeitungsmerkmalen zur Verfügung, noch zur Definition von Bearbeitungsabläufen, Strategien, Technologieparametern, Ressourcen und deren Alternativen. Die Konsequenz sind proprietäre Erweiterungen der APT-Schnittstelle (z.B. EXAPT) oder eigene, abgeschlossene Beschreibungsmodelle (z.B. CATIA NC, Open Mind). Demnach ist ein neutraler Austausch umfassender Planungsdaten heute nicht möglich [HAAS00] und lässt sich insbesondere mit den einfachen Befehlen der DIN Programme nicht darstellen. Dadurch stehen an der Werkzeugmaschine und in der NC-Steuerung keine Konstruktions- und Planungsdaten zur vereinfachten NC-Programmierung, Prozessüberwachung und -regelung zur Verfügung.

Ein zukünftiges Austauschformat der Arbeitsvorbereitung wird folglich Merkmals-, Operations-, Prozess- und Ressourcendaten enthalten müssen und durchgängig bis in die NC-Steuerung verfügbar sein. Sein Aufbau muss Erweiterungen zulassen, um neue Technologien und proprietäres, geschütztes Prozesswissen abbilden zu können. Beispiele für mögliche Erweiterungen sind Parameter für die Schnittaufteilung, Regelstrategien oder technologische Ziel- und Grenzwerte.

Funktionsgenaue Ablaufkodierung: Bei hochoptimierten Prozessen bieten elementare Steuerungsbefehle, wie die der DIN 66025, Vorteile. Der Programmier kann die Abfolge der Weg-, Technologie- und Schaltbefehle sekundengenau vorgeben, um Taktzeiten zu minimieren und die Steuerungsfunktionen zu synchronisieren. Zusatzfunktionen und externe Systeme, wie Beladeeinrichtungen, werden punktgenau über SPS Befehle zwischen Bewegungsanweisungen angesteuert. Mithilfe von Variablen lassen sich die exakte Aufspannlage, Werkstückabmaße, Korrekturparameter oder Daten zur Werkstückidentifikation flexibel kodieren und mit Messsystemen abgleichen. Diese Eigenschaften der DIN 66025 sind Grundforderungen der hochautomatisierten Massenfertigung und müssen auch weiterhin bedient werden.

Unterstützung des Werkers: Das Prozessverständnis des Werkers ist durch sein Aufgabenverständnis und die für ihn greif- und sichtbaren Elemente geprägt. Er denkt in Arbeitsschritten, in denen Bearbeitungsmerkmale nacheinander abgearbeitet werden, so dass final das zu fertigende Bauteil entsteht. Bei der Definition der Arbeitsschritte muss der Werker das Rohteil, die verfügbaren Maschinen und Werkzeuge sowie die Fertigteilgeometrie berücksichtigen [GLOC99, BROU00]. Die diskreten Steuerungsanweisungen des NC-Programms benötigt er nur, um den Ablauf zu optimieren, d.h. die Abarbeitung an das Systemverhalten anzupassen und Taktzeiten zu reduzieren. Die NC-Programmierung muss daher die Denkweise des Werkers unterstützen und ihm gleichzeitig optimierende Eingriffe auf Befehlsebene ermöglichen. Dazu müssen sich diese Elemente im NC-Programm wieder finden, um sie in den Programmier- und Anzeigesystemen grafisch anzeigen und dialoggeführt programmieren zu können.

Abarbeitung in der NC-Steuerung: Während CAx-Systeme Datensätze generieren, steuert die NC-Steuerung direkt Achsbewegungen und Maschinenfunktionen an. Eine „Zurück-Funktion“, um erkannte Fehler zu korrigieren oder Varianten auszuprobieren, gibt es nicht. Die programmierten Vorgaben müssen eindeutig zu interpretieren sein und sich mit dem Ziel einer unterbrechungs- und ruckfreien Bewegungsführung schnell einlesen und umsetzen lassen. Unternehmen mit zertifizierten Prozessen setzen zudem eine reproduzierbare Abarbeitung voraus. Sie fordern, dass sich Teile des NC-Programms und deren Abarbeitung gezielt

gegen Änderungen sperren lassen. Hierzu werden an der Steuerung die Lese- und Schreibrechte eingeschränkt. Ferner werden im laufenden Prozess die Override für Vorschub und Drehzahl abgeschaltet, um konstante Taktzeiten und die Einhaltung von Toleranzen sicherzustellen.

4.2. Anforderungen an die NC-Steuerung

Requirements on the numerical control

Die vielfältigen Anforderungen an den Informationsgehalt des NC-Programms führen zu NC-Programmierschnittstellen mit umfangreichen Befehls- und Datenmodellen sowie komplexen Formaten. Hierzu reichen die an die DIN 66025 angelehnten Interpreterfunktionen konventioneller NC-Steuerungen nicht aus, da sie ausschließlich auf das zeilenweise Einlesen elementarer mnemonischer Befehle ausgerichtet sind. Vor dem Hintergrund der Aufgabenstellung erfordert eine Prozessregelung auf der Basis der NC-Programmvorgaben, dass die Steuerung während der Bearbeitung den Zugriff auf diese Vorgaben zulässt, Eingriffe im Kontext der Bearbeitungsaufgabe auf Fehleingaben überwacht und Änderungen dynamisch in der Programmverarbeitung umsetzt.

Einlesen des NC-Programms: Wenn der Informationsgehalt des NC-Programms nicht mehr nur auf die Kodierung der zur Ausführung notwendigen Steuerungsbefehle und -parameter beschränkt ist, muss der NC-Interpreter in einem ersten Schritt die Steuerungsdaten von Zusatzinformationen unterscheiden können. Findet er im NC-Programm keine expliziten Weg-, Schalt- und Technologieanweisungen, so muss der Interpreter in der Lage sein, die höherwertigen Geometrie- und Operationsdaten zu interpretieren und aufzulösen. Im Fall einer offenen NC-Steuerung hat der Interpreter zusätzlich die Aufgabe, die anwenderspezifischen Programmdaten einzulesen und sie den Erweiterungen ablauf- und positionssynchron bereitzustellen.

Zugriff auf die NC-Programmvorgaben: Die NC-Steuerung liest das NC-Programm mit Vorlauf zur Bearbeitung ein, um Funktionen, wie die Werkzeugkorrektur oder die Geschwindigkeitsführung, befehlsübergreifend berechnen zu können. Folglich sind die aktuell und kurzfristig auszuführenden Programmvorgaben bereits in den Kern der Steuerung geladen. Mit dem Ziel, diese manipulieren zu können, reicht der Zugriff auf die NC-Programmdatei nicht aus. Vielmehr müssen der Werker oder ein Regelautomatismus kontrollierten Zugriff auf die intern verwalteten Programmvorgaben erhalten.

Dynamische Verarbeitung der Änderungen: Die Steuerung muss die geänderten Programmvorgaben erkennen, ihre Umsetzbarkeit bewerten, ggf. Fehlerstrategien auslösen oder den Vorlauf der Programmverarbeitung löschen und neu berechnen. Zuletzt muss die NC-Steuerung einen harmonischen Übergang von den aktuell ausgeführten auf die adaptierten Sollwerte generieren. Es darf nicht zu neuen Prozessstörungen, wie einem Ruck in der Bewegungsführung oder einer Kollision aufgrund von falsch synchronisierten Schaltfunktionen kommen.

Offenheit der NC-Steuerung: Mit dem Ziel, die Steuerungsfunktionalität anwenderspezifisch erweitern zu können, müssen die geforderten Mechanismen als Grundfunktionen bereitgestellt werden. Neben dem durchgängigen Datenzugriff auf die eingelesenen NC-Programmvorgaben sind ferner offene Schnittstellen zur Erweiterung der Programmierschnittstelle, der Programminterpretation, der Programmumsetzung sowie der lage- und zeitsynchronen Sollwertausführung vorzusehen.

4.3. Anforderungen an die Benutzerschnittstelle

Requirements on the user interface

Die einzelnen Steuerungsfunktionen müssen auf das Prozessverständnis und die Reaktionsfähigkeit des Anwenders und die Anforderungen einer automatischen Regelung abgestimmt sein.

Anzeige der Prozess- und Regelparameter: Prozessabweichungen werden durch den Vergleich der Ist-Daten mit den Sollvorgaben erkannt. Ausgehend von den Vorgaben des NC-Programms und dem aktuellen, realen Prozess muss die NC-Steuerung die Soll- und Ist-Daten bereitstellen und anwendergerecht aufbereiten. Die Art der Darstellung ist von dem Prozessverständnis des Anwenders abhängig. Der Werker kann aufgrund seiner Reaktionszeit und seinem merkmals- und arbeitsschrittorientierten Denken die Abarbeitung nicht im Lageregeltakt oder auf der Ebene einzelner Bewegungssegmente verfolgen. Ihm müssen die Soll- und Ist-Daten grafisch aufbereitet angezeigt werden. Eine automatisierte Regelung kann dagegen im Systemtakt bewerten und eingreifen. Sie erfordert Informationen auf der Ebene der steuerungsinternen Weg- und Schaltanweisungen.

Das Gesamtsystem muss ferner anzeigen, ob Grenzwerte und Regelstrategien im NC-Programm vorgegeben wurden oder Parameter für den Zugriff gesperrt sind. Die verfügbaren Eingriffsmöglichkeiten müssen dem Anwender ebenso transparent sein, wie deren Einfluss.

Zugriff auf die Regelparameter: Die Schnittstellen zum Eingriff in die NC-Programmvorgaben und zum Zugriff auf die Steuerungsfunktionen müssen eindeutig sein. Beispielsweise dürfen beim Bohren nur die technologisch relevanten, beeinflussbaren Größen angezeigt werden. Hierzu könnten die maximale Bohrtiefe je Zyklus, die Drehzahl oder der Vorschub gehören. Parameter anderer Bearbeitungsoperationen, wie die einer Taschenoperation, zur Korrektur von Schnitttiefe, Werkzeugüberdeckung oder Strategien zum tangentialen An- und Abfahren, dürfen nicht für eine Bohrung angeboten und angezeigt werden. Dies würden den Anwender zum einen verwirren und zum anderen wären die Eingaben durch die Steuerung nicht sinnvoll umsetzbar. Stattdessen muss der Werker durch eine situationsgerechte Anzeige von Soll- und Eingriffsparametern geführt werden. Es sind ihm geeignete Interaktionsmechanismen zum schnellen und effektiven Prozesseingriff bereitzustellen. Hierzu eigenen sich beispielsweise Schieber, Drehknöpfe und Auswahllisten, die in Abhängigkeit von der aktuellen Bearbeitungsaufgabe aktiviert und parametriert werden. Grafische Darstellungen der Bearbeitungssituation, der Variationsgrenzen und ihres zu erwartenden Effekts können den Anwender zusätzlich unterstützen.

Reaktionsverhalten des Systems: Für den Werker ist eine direkte Systemreaktion wichtig, um zu erkennen, dass er den Prozess kontrolliert. Anderenfalls gewinnt er den Eindruck, die Korrektur wäre nicht ausreichend gewesen und ohne den Effekt abzuwarten, verstärkt er den Eingriff oder bricht den Prozess ab. Die Reaktionszeit muss bei der Abfrage und der Umsetzung der Eingriffe berücksichtigt werden. Die Umsetzung ist dem Anwender anzuzeigen.

Im Fall gravierender Prozessstörungen oder Fehleingaben muss das System den Bearbeitungsprozess automatisch regeln oder sicher abbrechen. In jedem Fall muss der Anwender die finale Kontrolle behalten. Die Funktionen der konventionellen Eingriffsmechanismen, wie Not-Halt, Override, Programm- und Spindel-Stopp müssen aktiv bleiben.

Skalier- und Erweiterbarkeit: NC-Steuerungen werden technologieübergreifend eingesetzt. Um ihre Funktion an die spezifischen Anforderungen der Maschine, des Prozesses und des Anwenders anpassen zu können, bieten sie Maschinendaten zur Konfiguration und offene Schnittstellen zur Anpassung der SPS und der NC-Funktionen an. Diese Flexibilität, das Gesamtsystem entsprechend den eigenen Anforderungen zu erweitern und zu skalieren, muss auch für das zu entwickelnde Steuerungskonzept gelten.

5. Konzept einer aufgaben- und merkmalsorientierten Prozessregelung

Concept for a task and feature oriented process control

Zur Lösung der in Kapitel 0 zusammengefassten Anforderungen wird im Folgenden ein Gesamtkonzept entworfen, auf dessen Grundlage sich eine aufgaben-, merkmals- und operationsorientierte NC-Programmierung, Prozessverfolgung und bearbeitungsparallele Prozesskorrektur realisieren lassen. Im Kern des Konzepts stehen eine neuartige, die Aufgabe und den Prozess umfassend beschreibende NC-Programmierschnittstelle sowie Interaktions- und Steuerungsalgorithmen, die den laufenden Prozess analysieren, kontextorientiert Eingriffsoptionen anbieten und Programmänderungen dynamisch auswerten und zeitnah umsetzen.

Mit dem Ziel, die notwendigen Informationen bereitzustellen, um die Bearbeitungsaufgabe an der Maschine vollständig und situationsgerecht darstellen, optimieren und zur Prozessregelung auswerten zu können, wird in einem ersten Schritt eine aufgaben-, merkmals- und prozessorientierte **NC-Programmierschnittstelle** entwickelt. Diese NC-Programmierschnittstelle wird objektorientiert aufgebaut. Analog zu dem logischen Ablauf der NC-Programmierung wird in ihr gezielt zwischen Geometrien, Bearbeitungsaufgaben, Operationen und Steuerungsbefehlen unterschieden. Die Darstellungsform und der Umfang der Schnittstellenobjekte reichen aus, um ohne Konvertierungen und Zusatzdateien alle erforderlichen Informationen von der Geometrieübernahme aus der Konstruktion über die NC-Programmierung bis hin zur Prozesskorrektur und der Rückmeldung von Optimierungen auszutauschen, Bild 5.1. Zudem lässt sich die NC-Programmierschnittstelle strukturiert um eigene Objekte erweitern. Dieser Ansatz ist eine wesentliche Grundlage, um zukünftige Anforderungen aus der NC-Planung, Prozesstechnik oder aber neue Steuerungs- und Maschinenfunktionen über das NC-Programm mit Informationen bedienen zu können.

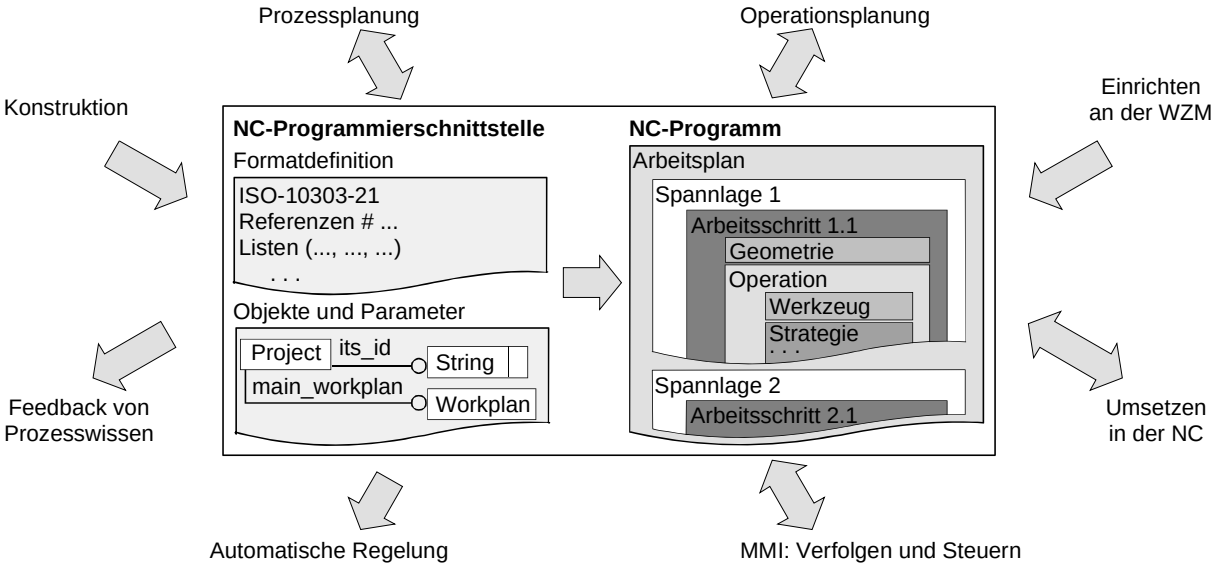


Bild 5.1: Aufgaben- und Prozessbeschreibung als durchgängige Informationsbasis

Manufacturing tasks and processes as input for a consistent information base

Ein ausführliches Programmbeispiel auf der Basis der nachfolgend entwickelten und beschriebenen NC-Programmierschnittstelle findet sich im Anhang 11.

Ausgehend von der strukturierten, erweiterbaren NC-Programmierschnittstelle ist ein durchgängiger Informationsaustausch von der NC-Planung bis zur Programmabarbeitung möglich. Anwendungsspezifische Zwischenformate, wie die heutiger CAM-Systeme, Postprozessoren

und Steuerungsprogramme, werden durch eine übergreifende Bearbeitungsbeschreibung ersetzt. Programmänderungen in der Planung oder Optimierungen an der Maschine stehen über einen zentrale Datenbasis allen Teilsystemen zur Verfügung, Bild 5.2. Die Risiken redundanter, aufgrund fehlender Verknüpfung nicht zwingend aktueller Datensätze werden umgangen. Ferner lassen sich die von der Maschine ausgeführten Abläufe, Funktionen und Verfahrwege im Programm protokollieren und für Wiederholbearbeitungen oder Neuplanungen nutzen.

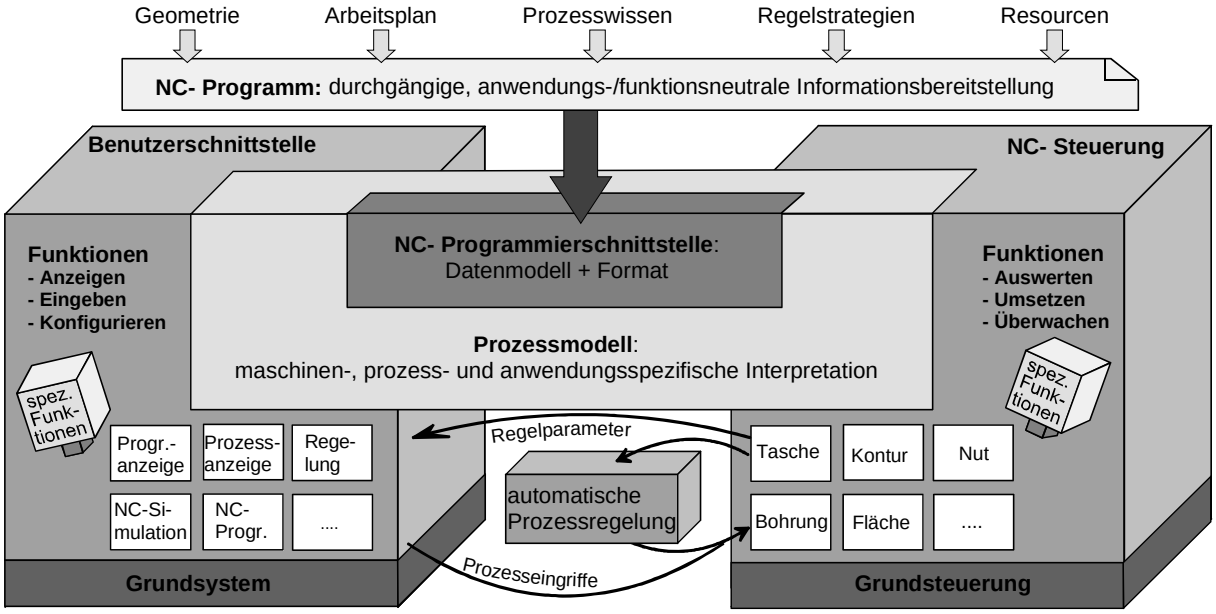


Bild 5.2: Aufbau des Gesamtkonzepts zur NC-programmbasierten Prozessregelung
Design of the concept for process manipulation based on the NC-program's data

Innerhalb des Gesamtkonzepts hat die **NC-Steuerung** die Aufgabe, die programmierte Bearbeitungsaufgabe zu interpretieren, ihre Daten auf das steuerungsinterne Prozessmodell abzubilden und sie danach in die zur Ausführung notwendigen Schalt- und Positionssollwerte umzusetzen, Bild 5.2 rechts. Neben Standardroutinen ist die NC-Steuerung offen für individuelle Erweiterungen, um Merkmale und Operationen spezialisiert umzusetzen oder zukünftige Prozessabläufe realisieren zu können. Zudem stellt die NC-Steuerung Schnittstellen zur Manipulation der eingelesenen NC-Programmdaten zur Verfügung. Hinter diesen Schnittstellen steht ebenfalls das aufgaben- und prozessorientierte Modell der NC-Programmierschnittstelle. Es erlaubt Prozesseingriffe auf unterschiedlichen Abstraktionsebenen, wie der Arbeitsschrittfolge, den Operations- und Technologieparametern oder der Bahnführung.

Die NC-Programmierung sowie die Benutzung und Verfolgung der vielfältigen Steuerungsfunktionen erfolgt mithilfe der **Benutzerschnittstelle**, Bild 5.2 links. Sie stellt dem Anwender intuitive, seinem Prozessverständnis entsprechende Hilfen zur NC-Programmierung, zum Einfahren neuer Prozesse, zur Programmausführung sowie zur Prozessverfolgung und -regelung bereit. Während der Bearbeitung zeigt sie das zu fertigende Merkmal, die Operationen oder die auszuführenden Parameter, Funktionen und Verfahrwege an. Zudem bietet sie kontextorientiert Eingriffsfunktionen an, um die Programmvorgaben online zu manipulieren.

Bevor in den Kapiteln 6 und 7 detailliert auf die Teilsysteme NC-Steuerung und Benutzerschnittstelle eingegangen wird, werden nachfolgend die übergreifenden Elemente des Konzepts beschrieben. Hierzu zählen die NC-Programmierschnittstelle zur Aufgabenbeschreibung (Kap. 5.1), die Kopplung von NC-Programmverarbeitung und Benutzerschnittstelle zur Prozessverfolgung und -manipulation (Kap. 5.2) sowie Lösungen zum Aufbau eines offen erweiterbaren softwaretechnischen Gesamtsystems (Kap. 5.3).

5.1. Aufgaben- und prozessbeschreibende NC-Programmierschnittstelle

NC-programming interface containing task and process descriptions

Die Vielzahl der in Kapitel 0 zusammengefassten Anforderungen setzt eine Programmierschnittstelle voraus, in der sich sowohl elementare Weg- und Schaltbefehle, als auch zusammenhängende Aufgaben- und Geometriebeschreibungen programmieren und durchgängig zwischen der Arbeitsvorbereitung, der Benutzerschnittstelle und der NC-Steuerung austauschen lassen. Auf der Grundlage des NC-Programms muss sich zudem der Bearbeitungsprozess für den Anwender verständlich darstellen lassen. Die operativen und technologischen Programmelemente müssen den Prozess effektiv beschreiben und so strukturiert sein, dass sie auch während der Abarbeitung in der NC-Steuerung zur Prozessverfolgung und -manipulation offen gelegt werden können. Für die Funktionserweiterung zur Überwachung und Regelung sind Eingriffsgrenzen, Regelstrategien und Grenzwerte im NC-Programm vorzuhalten.

Befehlsorientierte Programmierschnittstellen, wie die auf der DIN 66025 basierenden Lösungen der Steuerungshersteller, genügen diesen Anforderungen nicht. Ihr Befehlsumfang, d.h. die zur Verfügung stehenden Programmelemente reichen für die Beschreibung der Planungs- und Steuerungsinformationen nicht aus. Das Gesamtkonzept baut daher auf einem Datenmodell auf, dessen Kern standardisiert ist und das sich strukturiert erweitern lässt. Im einfachsten Fall können die einzelnen Elemente dieses Datenmodells elementare, den steuerungsinternen Weg-, Schalt- und Technologiefunktionen entsprechende Befehle und Parameter sein. Somit ließe sich der Inhalt eines heutigen NC-Programms weiterhin verlustfrei darstellen (vergl. Kapitel 6.2.2, Bild 6.10). Darüber hinaus sind komplexe, zusammenhängende Aufgaben-, Geometrie- und Operationsbeschreibungen enthalten, die zum einen den Anforderungen nach einem durchgängigen, bidirektionalen Datenaustausch mit der Arbeitsvorbereitung gerecht werden und zum anderen die aufgaben- und merkmalsorientierte Denkweise des Programmierers unterstützen sowie der Steuerung Informationen zur autonomen Aufgabenausführung, zur Prozessüberwachung und zur Prozessregelung bereitstellen.

5.1.1. Formaler Aufbau der Programmierschnittstelle

Formal structure of the NC-programming interface

Da der Informationsgehalt dieser neuen NC-Programmierschnittstelle über die reine Befehlskodierung hinausgeht und komplexe Aufgaben- und Geometriebeschreibungen abzubilden sind, lassen sich die mnemonischen Codes und der sequentielle Aufbau der DIN 66025 nicht übernehmen. Vielmehr muss die Beschreibungssprache zur NC-Programmierung in der Lage sein, strukturierte Konstruktions-, Aufgaben- und Prozessmodelle und deren Verschachtelung abzubilden. Die Programmierschnittstelle wird daher nicht mehr aus einzelnen Befehlsworten aufgebaut, sondern enthält eine objektorientierte, hierarchische Beschreibung der Bearbeitungsaufgabe. Die im NC-Programm gespeicherte Aufgabenbeschreibung wird ausgehend von einem zentralen Datenelement, dem „*Project*“, über die Bearbeitungsaufgabe, die darin enthaltene Geometrie- und Operationsbeschreibung, bis hin zu den expliziten Parametern, Verfahrwegen und Maschinenfunktionen schrittweise detailliert. In Bild 5.3 ist der Aufbau eines derartigen Datenmodells und des daraus resultierenden NC-Programms dargestellt. Jedes Kästchen, bzw. jede Programmzeile entspricht einem Datenelement. Die von den Elementen ausgehenden dünnen Linien verweisen auf die Attribute des Elements. Im NC-Programm werden sie sequentiell hinter dem Elementnamen als Abfolge von Texteinträgen, Zahlenwerten und Verweisen kodiert. Beispielsweise beschreibt das Element #2 „*Workplan*“ den abzuarbeitenden Arbeitsplan. Zu seinen Attributen gehören unter anderem sein Name („*its_id*“), eine Liste der auszuführenden Arbeitsschritte und NC-Funktionen („*its_elements*“) sowie die Aufspannung („*its_setup*“). Der erste Eintrag in der Arbeitsschrittliste verweist auf die Programmmarke „*#10*“. D.h. zuerst wird die Marke gesucht und der dort kodierte Arbeits-

schritt #10 ausgeführt. Dieser enthält wiederum Attribute, die u.a. das zu bearbeitende Merkmal, eine Tasche „*#11*“, und die daran auszuführende Operation „*#12*“ referenzieren. Mithilfe der Datenelemente lässt sich so schrittweise die gesamte Bearbeitung ausgehend von einem übergeordneten Ablaufplan, über einzelne Arbeitsschritte, Operationen und Bearbeitungsmerkmale bis hin zu den final auszuführenden Weg-, Schalt- und Technologiebefehlen programmieren. Der Aufbau und die Datenelemente der NC-Programmierschnittstelle werden nachfolgend detaillierter beschrieben.

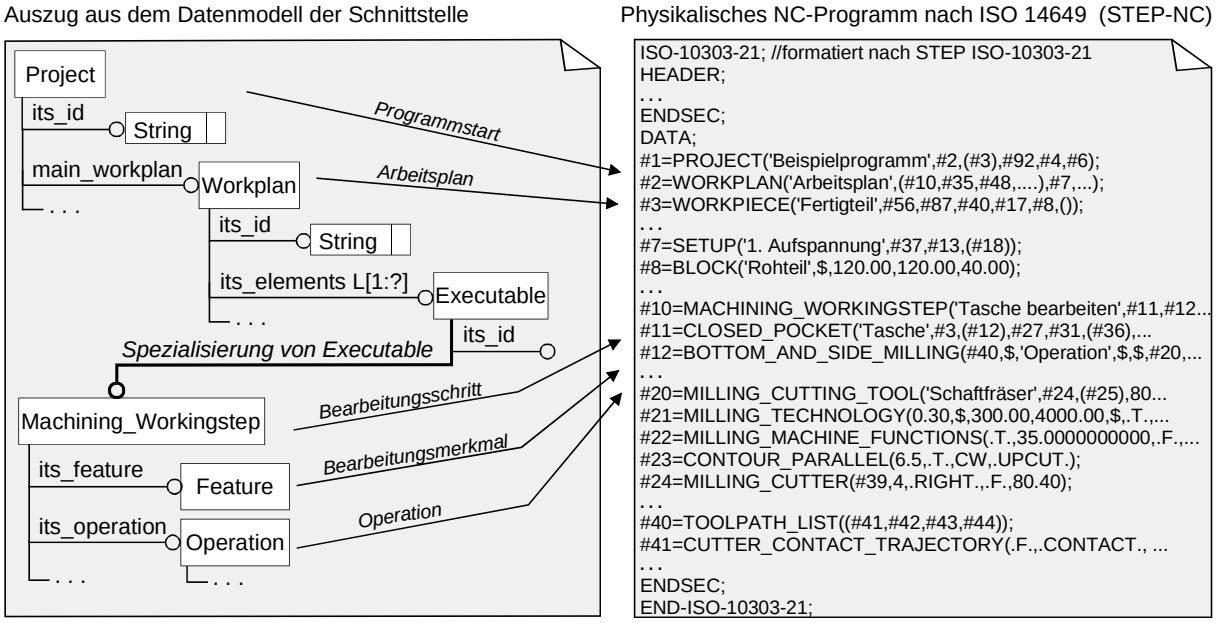


Bild 5.3: Strukturierte, zusammenhängende Aufgabenbeschreibung im NC-Programm
Structurized and linked task description inside the NC-program

Analog zu dem Programmierschlüssel der DIN 66025, in dem der Aufbau und die Bedeutung der mnemonischen Codes festgehalten sind, müssen für das zu konzipierende Datenmodell ebenfalls die Bedeutung und der formale Aufbau der verfügbaren Elemente definiert werden. Die im Fall der vorliegenden Arbeit entwickelten Datenelemente sind größtenteils in die Normierung der ISO 14649 eingeflossen und dort dokumentiert worden [ISO14649-1, -10, -11]. Daher werden nachfolgend nur ausgewählte, für das Verständnis der Arbeit besonders relevante Datenelemente detailliert erläutert. Auszüge aus dem hier entwickelten und international standardisierten Grund- und Fräsmodell finden sich im Anhang 11.1 der Arbeit.

Die Struktur und die Inhalte der NC-Programmierschnittstelle werden mithilfe einer grafischen Beschreibungssprache definiert. Das daraus resultierende Datenmodell wird in eine maschinenlesbare Textform konvertiert und final in eine softwaretechnisch nutzbare Datenstruktur oder Klassenbibliothek umgesetzt (Bild 5.4). Aufgrund des generischen Aufbaus und der einfachen Symbolik bzw. Modellierungssprache lässt sich die Programmierschnittstelle durch Dritte um deren spezifische Programmelemente erweitern. Ein Compiler setzt das geänderte Modell erneut in eine Bibliothek um. Diese wird von der Steuerungssoftware eingebunden und stellt ihr die zur Speicherung und Verarbeitung des NC-Programms notwendigen Datenstrukturen bereit. Dieser Ansatz erlaubt es, die Schnittstelle strukturiert um neue Elemente zu erweitern und erprobte Lösungen später in die Normung zu übernehmen, so dass die Austauschbarkeit gewährleistet ist. Eine Anpassung der bestehenden Steuerungsfunktionen ist nur dann notwendig, wenn Elemente aus dem Modell entfernt oder geändert werden. Die spezifisch hinzugefügten Datenelemente werden mithilfe der Bibliothekfunktionen in die NC-Steuerung eingelesen und dort an die durch den Entwickler ergänzten Interpretationsfunktionen weitergeleitet. Dieser Ansatz erlaubt es, die NC-Programmierschnittstelle um individuelle

Datenelemente zu erweitern, die Erweiterungen in die Steuerung einzulesen und dort den spezifischen Funktionserweiterungen zur Verarbeitung bereitzustellen. Diese Lösung entspricht der Anforderung, die Programmierschnittstelle flexibel an zukünftige Entwicklungen anpassen zu können.

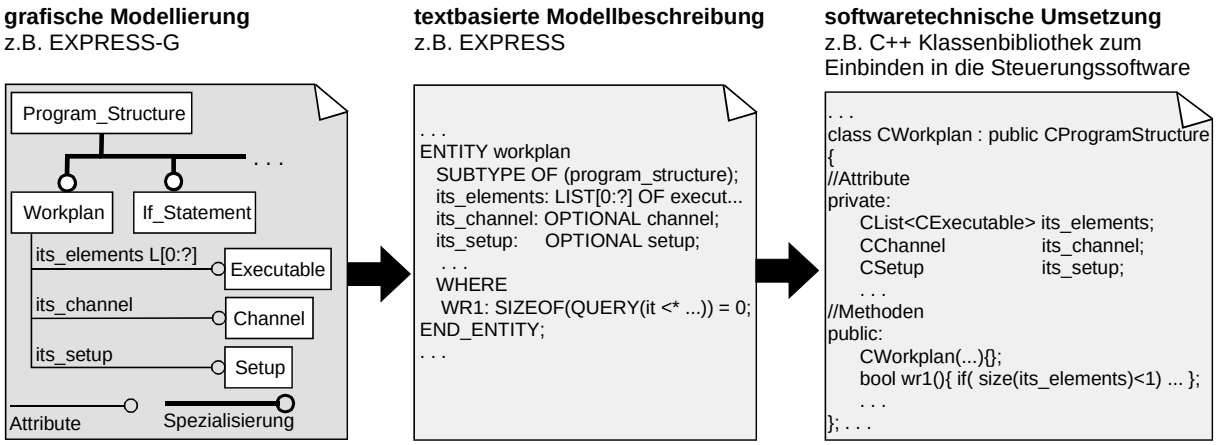


Bild 5.4: Modellieren der NC-Programmierschnittstelle und Umsetzen in eine Bibliothek
Modeling of the NC-programming interface and realization in a software library

In der Industrie haben sich verschiedene Beschreibungssprachen zur Modellierung und physikalischen Darstellung komplexer Datenstrukturen etabliert. Im Kontext der Geometrie- und Produktdatenbeschreibung nach STEP sind die Extensible Markup Language (XML), und EXPRESS hervorzuheben. Der Vorteil von XML liegt in seinem einfachen, dem HTML-Standard zur Beschreibung von Webseiten ähnlichen Aufbau. Darüber hinaus finden sich viele Applikationen und Bibliotheken, die die Arbeit mit XML Datenmodellen und -Datensätzen unterstützen. Daher nutzen Steuerungshersteller XML zur Beschreibung und zum Abspeichern ihrer Maschinen-, Steuerungs- und Werkzeugdaten [BOSC04b].

Die Beschreibungssprache EXPRESS ist ein Teil der STEP Normenreihe [ISO10303-11] und wird verwendet, um die einzelnen Datenmodelle des STEP Standards zu beschreiben. EXPRESS erlaubt die Modellierung eines hierarchischen Datenmodells mit komplexen Vererbungsstrukturen. Eine solche Vererbung stellt eine Spezialisierung dar, bei der ein Element von einem anderen abgeleitet wird, indem dessen Eigenschaften, d.h. die Attribute und Funktionen übernommen und spezifisch erweitert werden. Ausgehend von der Geometrie- und Produktbeschreibung nach STEP sind sowohl kommerzielle als auch freie Softwarebibliotheken zum Ein- und Auslesen von EXPRESS Modellen sowie deren Umsetzung in Softwarestrukturen verfügbar [SCHU00].

Auf der Grundlage der STEP Normenreihe und deren Kompatibilität lassen sich XML-Schemata in EXPRESS Modelle überführen, als auch ihre Datenkodierungen ineinander konvertieren. Beide Ansätze integrieren daher die Produktbeschreibung nach STEP und eignen sich für die durchgängige Aufgabenbeschreibung von der Konstruktion bis zu der hier zu entwickelnden Programmierung der NC-Bearbeitung. Da EXPRESS und die physikalische, textbasierte Kodierung nach STEP Part 21 [ISO 10303-21] ein kompakteres Format anbieten, ist ihre datentechnische Verarbeitung schneller und wird aufgrund der zeitkritischen Zugriffsanforderungen der NC-Technik dem nachfolgenden Konzept zugrunde gelegt.

5.1.2. Datenelemente der NC-Programmierschnittstelle

Data elements of the NC-programming interface

Das Datenmodell der NC-Programmierschnittstelle setzt auf Vorarbeiten zur splinebasierten Freiformflächenprogrammierung [MATR97], der Trennung von Fertigungsgeometrie und Bearbeitungsschritt [OPTI97, SCHU00] sowie der featureorientierten Aufgabenbeschreibung des Sonderforschungsbereich 368 „Autonome Produktionszelle“ [APZ02] auf. Aus diesen Ansätzen wurden die Unterscheidung zwischen Arbeitsschritt und Bearbeitungsmerkmal sowie die Kurven- und Flächenbeschreibungen analog dem STEP Standard übernommen.

Um eine durchgängige Integration in die STEP Produktmodellierung zu erreichen, wurden in einem ersten Schritt die bestehenden Modellansätze überarbeitet und insbesondere die stark zyklenorientierten Featuredefinitionen durch eine Neudefinition auf der Grundlage der *Manufacturing Feature* [ISO10303-224] mit der STEP Normenreihe abgeglichen [WOLF00]. Damit wurde die Voraussetzung geschaffen, die vollständige Geometriebeschreibung durchgängig aus dem CAD bis in die NC-Steuerung auszutauschen und in der NC-Steuerung zur Kollisionsüberwachung oder autonomen, prozessangepassten Bahnaufteilung zu verarbeiten [WECK03].

Zudem wurde präzise zwischen der Geometrie und der Bearbeitungsoperation unterschieden. Anstelle von geometriespezifischen Operationsschritten, z.B. „*Pocket Rough Milling*“ [BROU00], wurden separate Datenelemente zur Definition der zu fertigenden Geometrie (*Machining Feature*), der Arbeitsschrittfolge (*Workingstep*) und den anzuwendenden Operationen (*Machining* O*peration*) eingeführt [WOLF01, SCHL02]. An der Maschine erlaubt diese Struktur es dem Werker, die Bearbeitung so darzustellen und zu programmieren, dass sie seinem individuellen Technologiewissen entspricht und an die verfügbaren Ressourcen und Spezifika der jeweiligen Maschine durch den gezielten Austausch der Operationsdatensätze angepasst wird. Beispielsweise kann eine zylindrische Aussparung alternativ gebohrt, ausgespindel oder zirkular gefräst werden. Ebenso lässt sich in der Planung eine erprobte Operationsdefinition zur Bearbeitung ähnlicher Geometriemerkmale als Erfahrungswissen speichern und wieder verwenden.

Durchgängige Unterstützung der NC-Programmierung

Die Strukturierung der NC-Programmierschnittstelle in Geometrie, Aufgabe und Operation spiegelt die Teilaufgaben der NC-Programmierung von der Geometrieübernahme über die Prozess- und Operationsplanung, bis zur Möglichkeit der expliziten, maschinenorientierten Vorgabe von Schalt-, Weg- und Technologiebefehlen wider, Bild 5.5. Zudem unterstützt sie den Werker an der Maschine, um gezielt auf die Geometrie oder die Operation zuzugreifen, ohne die Aufgabe in Form von Steuerungs- und Maschinenfunktionen abstrahiert verfolgen und beschreiben zu müssen. Auf der Grundlage dieses Konzepts wurde die sogenannte STEP-NC Programmierschnittstelle überarbeitet, so dass sie 2004 international standardisiert [ISO 14649-10] und anschließend technologiespezifisch für das Fräsen [ISO14649-11], Drehen [ISO14649-12] und Erodieren [ISO14649-13] erweitert werden konnte.

In der Geometrieübernahme werden ausgehend von dem Fertigteil (*Workpiece* #3 in Bild 5.3 und Bild 5.5) die zu fertigenden 3D-Flächen (*Region*) und prismatischen Merkmale (*Drill Hole, Slot, Pocket* #11,…) aus dem CAD Modell übernommen oder neu abgeleitet. Zusätzlich werden Hilfsgeometrien eingeführt, um die Aufmaße des Rohteils (*Rawpart* #8) abzuarbeiten und Bearbeitungsmuster oder zusammenhängende Merkmale zu gliedern (*Pattern, Compound Feature*). Das Ergebnis ist eine zusammenhängende geometrische Beschreibung des Rohteils, der abzuarbeitenden Flächen und Volumina sowie die Übernahme der STEP-Fertigteilgeometrie (*Feature, Advanced Shape Representation*) aus dem CAD. Diese Informationen bilden die Grundlage für eine werkergerechte grafische Darstellung der Geometrie, die weitere Bearbei-

tungsplanung sowie die Möglichkeit Bearbeitungsbahnen und Bearbeitungsquerschnitte zu berechnen sowie mit der Werkzeugbeschreibung Kollisionen zu überwachen.

Um alle Geometriemerkmale verlustfrei übernehmen und beschreiben zu können, ist eine Vielzahl, zum Teil komplexer geometrischer Beschreibungen erforderlich. In Abhängigkeit von der geforderten Systemfunktionalität wird der Umfang der Datenelemente über Konformitätsklassen eingeschränkt. Die Geometriebeschreibung lässt sich implizit in Form von reinen Verfahrwegen (*Toolpath* #40) oder explizit als zu fertigende Feature (*Feature* #11) und Freiformflächen stufenweise skalieren. Dieser Ansatz ermöglicht es dem Programmierer, die Bearbeitung präzise durch Verfahrbewegungen vorzugeben oder aber der Steuerung eine Operation vorzugeben nach der sie selbstständig die zur Fertigung notwendigen Bahnen errechnet. Der Steuerungshersteller integriert in Abhängigkeit von den seinem Aufgabenspektrum entsprechenden Operationen und Featuren seine Algorithmen zur Bahn-, Feature-, Kurven-, Flächen- und Geometrieauswertung.

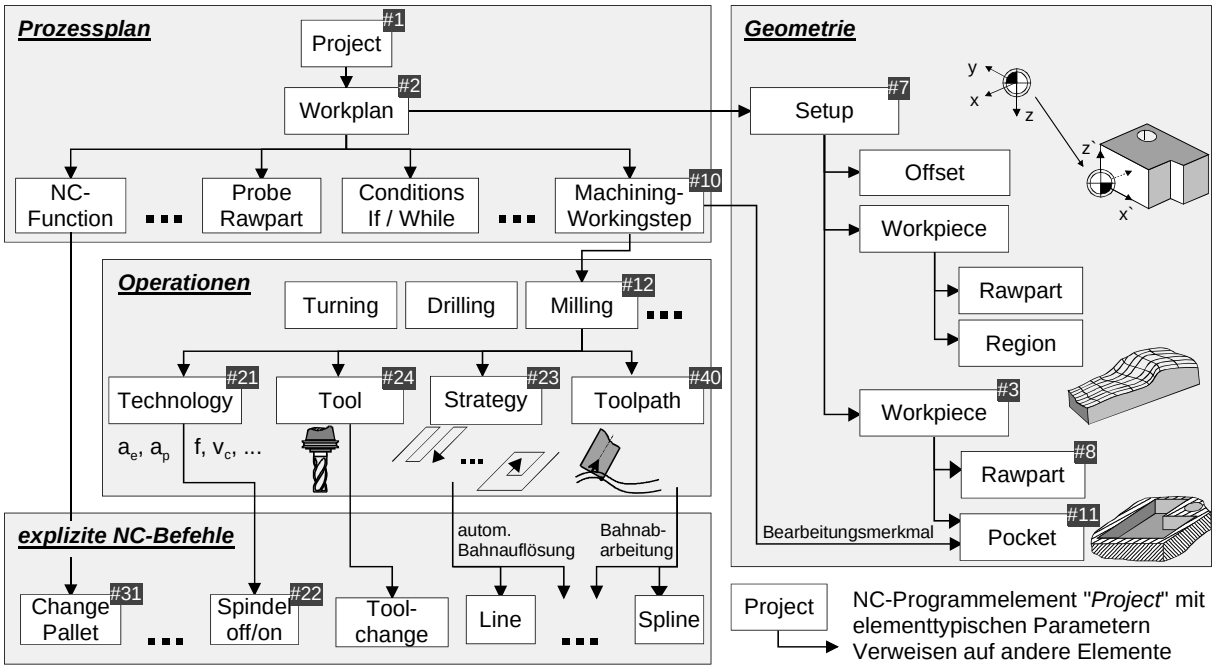


Bild 5.5: Durchgängige Aufgabenbeschreibung von der Geometrie bis zum NC-Befehl (Referenzen #... beziehen sich auf das Beispiel in Bild 5.3.)
Consistent task description from geometry to operations to NC-commands

Während sich der größte Teil der entwickelten Geometrieelemente in der ISO 14649 wieder findet, wurde ein Teil des Konzepts nicht übernommen. Grund hierfür ist die Ausrichtung der für die ISO 14649 verantwortlichen Normungsgruppe ISO/TC184/SC1[1]. Ihr Fokus lag auf der Programmabarbeitung in der NC-Steuerung. In der Folge lassen sich die CAD-Fertigteilgeometrie und die von der Arbeitsplanung ergänzten Hilfsgeometrien nicht unterscheiden. Beispielsweise ist eine zu fräsende Spannfläche datentechnisch nicht von einer Ebene am späteren Fertigteil zu unterscheiden. Im Datenmodell werden die fertigungstechnischen Bedeutungen und Abhängigkeiten nicht abgebildet. Nach einer Programmänderung muss der gesamte Datensatz durchsucht werden, um Wechselwirkungen mit anderen Merkmalen zu erkennen. In umgekehrter Richtung ist nicht erkennbar, ob die Geometrie zur Vereinfachung der Fertigung geändert werden darf oder ob sie fest vorgegeben ist. In der NC-Steuerung wird

[1] Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC 1, *Physical device control*

so die Bewertung einer Konturverletzung erschwert und es kann nicht in jedem Fall autonom entschieden werden, ob eine Korrektur der Technologieparameter und Bearbeitungswege zulässig ist. An dieser Stelle hätten boolesche Operationen oder zusätzliche Attribute in Kombination mit der im Arbeitsplan definierten Abarbeitungsfolge die Zusammenhänge eindeutig darstellen können: „*its_auxiliary_geometry*“ als eine Liste der zuvor zu bearbeitenden Hilfsgeometrien und „*its_originating_geometry*“ als Bezug zur Zielgeometrie, die selber wiederum aus Hilfsgeometrien oder den finalen Konstruktionsmerkmalen bestehen kann, Bild 5.6. Dieser Ansatz würde dem Werker gezielte Eingriffe in die Hilfsgeometrien erlauben, um beispielsweise die Bearbeitung an geänderte Rohteilabmaße, Werkzeuge oder Bearbeitungsstrategien anzupassen. Gleichzeitig wäre die Fertigteilgeometrie eindeutig erkennbar und ließe sich durch das NC-Programmiersystem oder die NC-Steuerung vor Manipulationen sperren.

In den Standard wurde anstelle dieser direkten Verknüpfung von Hilfs- und Fertigteilgeometrie nur eine Beschreibung des geometrischen Effekts eines Bearbeitungsschritts vorgesehen: „*in_process_geometry*“. Aus der Sicht der Entwickler von Planungsapplikationen hat die Prozessgeometrie den Vorteil, dass das geometrische Ergebnis auch ohne das Wissen über die zur Abarbeitung eingesetzten NC-Algorithmen und Werkzeuge direkt ausgelesen und dargestellt werden kann [ISO 14649-10]. Jedoch bedeutet die zusätzliche Geometriebeschreibung eine redundante Information gegenüber der Operations- und Merkmalsbeschreibung im Arbeitsschritt. Welche Information soll die NC-Steuerung auswerten? Wenn beispielsweise der in der Operation vorgegebene Werkzeugradius größer ist, als der kleinste Eckenradius der Prozessgeometrie, so müsste die Steuerung eine Diskrepanz erkennen, die Bearbeitung stoppen und den Anwender zur Programmkorrektur auffordern.

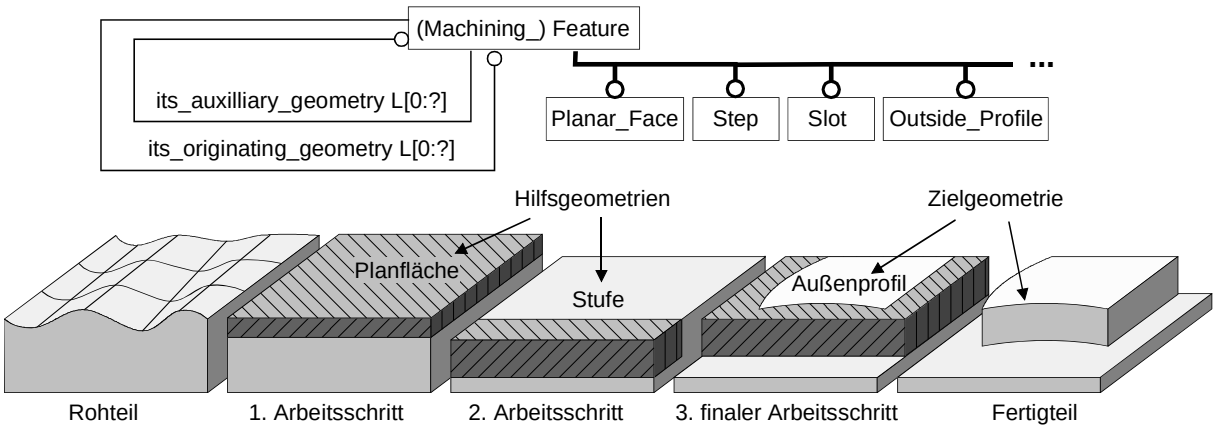


Bild 5.6: Bezug zwischen Hilfs- und Fertigteilgeometrie
Relation between auxiliary machining geometry and final part geometry

Da diese Arbeit die Aufgaben an der Maschine und die Eingriffe in den laufenden Bearbeitungsprozess fokussiert, werden im Folgenden die Abhängigkeiten zwischen Konstruktions- und Bearbeitungsmerkmalen nicht weiter betrachtet. Um jedoch darüber hinaus die Konstruktion und die Bearbeitungsplanung stärker zu koppeln und Änderungen direkt abgleichen zu können, sollte der obengenannte und in Bild 5.6 skizzierte Ansatz verfolgt werden.

Der Prozessplan strukturiert die Bearbeitung in Form von einzelnen Bearbeitungsschritten (*Workingstep*) und Funktionen (*NC-Function*), deren Abarbeitungsreihenfolge in Arbeitsplänen (*Workplan*) vorgegeben wird (vgl. #10, #31, #2 in Bild 5.3 und Bild 5.5). Zudem lassen sich die Arbeitspläne wiederum durch Arbeitspläne für die einzelnen Bearbeitungsprozesse und Spannlagen untergliedern. Das Setup-Attribut eines Arbeitsplans definiert dessen Aufspannung gegenüber der Werkzeugmaschine oder aber dessen Nullpunktverschiebung in Relation zu einem anderen, ihm übergeordneten Arbeitsplan. Dabei können durchaus mehrere Werkstücke parallel bearbeitet werden. Es ist über die Arbeitspläne eindeutig erkennbar,

wann die Spannlage gewechselt wird und ggf. eine andere NC-Maschine die weitere Bearbeitung übernimmt. Diese Struktur erlaubt es der nachfolgenden Operationsplanung, dem Werker und letztlich den Algorithmen der NC-Steuerung, die Bearbeitung logisch zu untergliedern, nachzuvollziehen und feature- sowie prozessorientiert zu optimieren.

In der Operationsplanung wird die Ausführung technologisch detailliert und es werden die einzusetzenden Werkzeuge (*Tool* #22 in Bild 5.3 und Bild 5.5), Strategien (*Strategy* #23) und Technologieparameter (*Technology* #21) definiert. Mit diesem Strukturierungsansatz wird der Operationsdefinition nach Eversheim entsprochen. Demnach ist eine Operation durch die Bearbeitung eines in sich abgeschlossenen Bearbeitungsmerkmals auf einer Maschine, in einer Aufspannlage und mit einem Werkzeug definiert [EVER97].

Detaillierte Ablauf-, Merkmals- und Technologiekodierung

Analog zu den Konformitätsklassen der Geometriebeschreibung lässt sich auch der Detaillierungsgrad der Operationsplanung an die Leistungsfähigkeit der NC-Steuerung anpassen. Dazu werden zum einen die Operationsdefinitionen technologiespezifisch unterteilt (Fräsen, Drehen, EDM etc.). Zum anderen kann die Ausführung der Bearbeitung der Steuerung in Form von expliziten Weg-, Technologie- und Schaltbefehlen vorgegeben werden (*Toolpath* mit wegsynchroner *Machine-Function* und *Technology*) oder aber die Steuerung generiert die erforderlichen Befehle autonom, indem sie die Bearbeitungsmerkmale entsprechend den Strategie-, Technologie- und Werkzeugdefinitionen des NC-Programms auflöst. Die verschiedenen Migrationsstufen des Datenmodells wurden auf der Basis universitärer und industrieller CAM, WOP und NC Prototypen im Rahmen des internationalen STEP-NC Projektes positiv erprobt (vgl. Kapitel 8). Dabei zeigte sich, dass sich die Mehrzahl der Feature und Operationen auf bestehende NC-Zyklenbibliotheken abbilden lassen und so von der NC-Steuerung ohne die aufwändige Entwicklung neuer Algorithmen abgearbeitet werden können [STEP01].

Eine Kombination aus expliziten Bearbeitungsvorgaben bei einer gleichzeitigen Operationsdefinition ist ebenfalls möglich. Sie schafft die Grundlage, um einerseits den Bearbeitungsablauf hochgradig optimiert zu beschreiben und andererseits an der Steuerung oder im laufenden Prozess flexibel auf die in der Operation kodierte Prozessführung zu wechseln. Ferner ist es ausgehend von der Operationsdefinition möglich, das Bearbeitungsziel zu beurteilen und alternative „Vorschläge“ zu generieren. Der Werker oder eine Regelung können komfortabel zwischen den Alternativen auswählen und den Prozess schnell und wirkungsvoll manipulieren. Das Datenmodell kann damit sowohl die Programmabläufe konventioneller, an die DIN 66025 angelehnter NC-Programmierschnittstellen, als auch Aufgabenbeschreibungen für eine flexible NC-Programmierung an der Maschine oder eine autonomen Umsetzung in der NC-Steuerung abbilden.

Gleichwohl steht dieser Ansatz in der Kritik, weil der Werker an der Maschine den Programmcode aufgrund der komplexen Syntax sowie der vielfältigen Programmelemente nicht mehr händisch editieren kann. Der Steuerungshersteller Siemens schlug daher das Festhalten am DIN-Code vor und dessen Erweiterung um teilstandardisierte Textblöcke. Das NC-Programm ließe sich so feature-, aufgaben- und operationsorientiert strukturieren [MENZ03].

Jedoch sind die hier konzipierten und in die Normung eingebrachten Elemente nicht als eingefügte Textinformation zu verstehen. Sie sind vielmehr Teil einer Bearbeitungsaufgabe und beschreiben diese vollständig. Prozessschritte, Merkmale und Operationen sind eindeutig getrennt, und nicht, wie in Bild 5.7 vorgeschlagen durchweg als ‚FEATURE’ vereinheitlicht. Erst eindeutige Elemente stellen Zusatzinformationen bereit, die heute nicht in der NC-Programmierung und insbesondere nicht an der Maschine verfügbar sind. Diese Programmelemente schaffen die Basis, um weit reichende Programmierhilfen, Simulationen, Algorithmen zur Fehlersuche, Eingriffsmethoden und Systeme zur Prozessüberwachung oder -regelung zu realisieren. Die Bereitschaft, den einfachen DIN-Code aufzugeben und ein rein über Softwareapplikationen zu bearbeitendes Programmformat zu akzeptieren, wird maßgeblich von

der Qualität und den zusätzlichen, den Anwender unterstützenden Funktionen dieser Applikationen abhängen. Ausgehend von einem standardisierten Format und Grundmodell könnten die Systeme herstellerübergreifend eingesetzt werden, sich funktionell ergänzen und gleichsam werden ausgehend von einer offenen Struktur Erweiterungen zur Realisierung zukünftiger oder individueller Spezialfunktionen möglich.

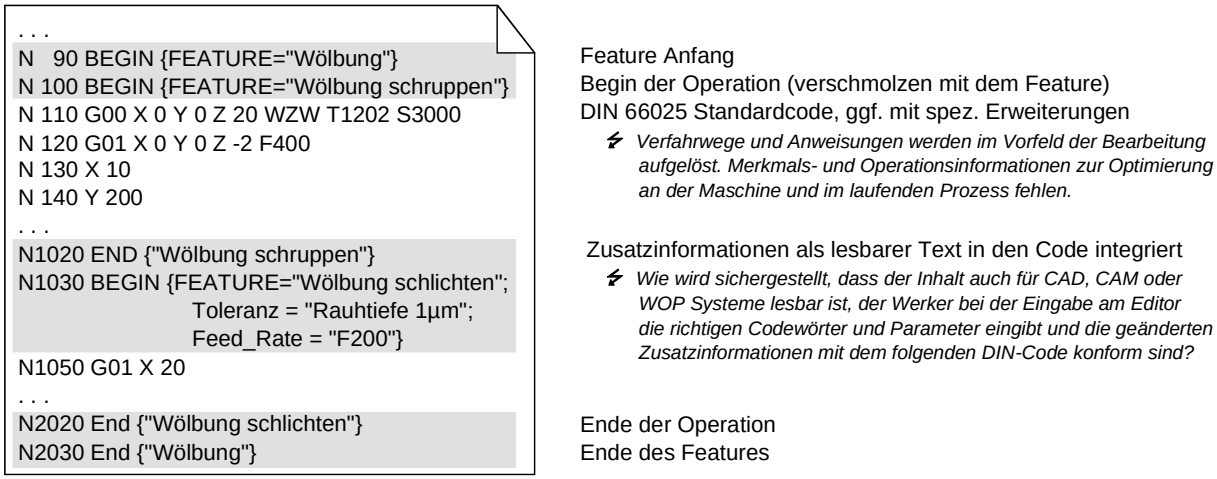

. . .
N 90 BEGIN {FEATURE="Wölbung"}
N 100 BEGIN {FEATURE="Wölbung schruppen"}
N 110 G00 X 0 Y 0 Z 20 WZW T1202 S3000
N 120 G01 X 0 Y 0 Z -2 F400
N 130 X 10
N 140 Y 200
. . .
N1020 END {"Wölbung schruppen"}
N1030 BEGIN {FEATURE="Wölbung schlichten";
Toleranz = "Rauhtiefe 1µm";
Feed_Rate = "F200"}
N1050 G01 X 20
. . .
N2020 End {"Wölbung schlichten"}
N2030 End {"Wölbung"}
Feature Anfang
Begin der Operation (verschmolzen mit dem Feature)
DIN 66025 Standardcode, ggf. mit spez. Erweiterungen
Verfahrwege und Anweisungen werden im Vorfeld der Bearbeitung aufgelöst. Merkmals- und Operationsinformationen zur Optimierung an der Maschine und im laufenden Prozess fehlen.
Zusatzinformationen als lesbarer Text in den Code integriert
Wie wird sichergestellt, dass der Inhalt auch für CAD, CAM oder WOP Systeme lesbar ist, der Werker bei der Eingabe am Editor die richtigen Codewörter und Parameter eingibt und die geänderten Zusatzinformationen mit dem folgenden DIN-Code konform sind?
Ende der Operation
Ende des Features

Bild 5.7: Feature-orientierte NC-Sprachelemente als Ergänzung der DIN 66025 [MENZ03]
Siemens proposal to extend the existing G-Code by textual Feature descriptions

5.1.2.1. Datenelemente zur Prozessüberwachung und -regelung

Code elements for process monitoring and process control

Neben der reinen Bearbeitungsbeschreibung muss das Datenmodell die Eingangsinformationen für die Prozessüberwachung und -regelung, d.h. für die spezialisierten Zusatzfunktionen der Steuerung liefern. Entgegen den Feature- und Flächenbeschreibungen aus der Konstruktion oder den steuerungsübergreifend ähnlichen NC-Zyklen zur merkmalsorientierten NC-Programmierung stehen derartige Beschreibungen für die Überwachung und Regelung nicht zur Verfügung. Beispielsweise sind im Brankamp-Artis-Prometec-Standard-Interface nur Daten- und Funktionsschnittstellen auf der Ebene der SPS definiert (vgl. Kapitel 2.3.3). Erwähnenswert an dieser Schnittstelle ist die Idee, Bearbeitungsinformationen parallel zum Programmablauf in Form eines „Schnittkode“ und „Werkzeugkode“ bereitzustellen. Somit kann eine Prozessüberwachung oder -regelung diese Informationen auswerten und gezielt auf die Bearbeitungssituation eingestellt werden.

Idealerweise lassen sich die für die Überwachung und Regelung erforderlichen Parameter bereits in der NC-Programmierung anhand des geplanten Sollprozesses und der möglichen Prozessalternativen definieren und im NC-Programm kodieren. Dazu wird der BAPSI-Ansatz aufgegriffen und in Form von zusätzlichen Attributen in das Datenmodell der NC-Programmierschnittstelle übertragen. Von Vorteil ist dabei der hierarchische, die Bearbeitungsaufgabe schrittweise präzisierende Aufbau des Datenmodells. Ausgehend von der Aufspannung über die Abarbeitungsfolge, die Operationen, die Technologieparameter und letztlich die explizit vorgegebenen Weg- und Schaltbefehle lassen sich in allen Ebenen zusätzliche Überwachungs- und Regelungsparameter ergänzen.

In der Definition der Spannlage (*Setup*) werden neben den Positionen der zu bearbeitenden Bauteile weitere, für eine Lagekorrektur (*Position Offset*) oder die Generierung kollisionsfreier Bewegungen wichtige Zusatzinformationen abgespeichert, vgl. Bild 5.5. Hierzu zählen globale, arbeitsschrittübergreifende Sicherheitsebenen (*Global Security Plane*), gesperrte Be-

reiche (*Restricted Area*), die Rohteilabmessungen (*Bounding Box*) sowie die Spannstellen und Spannmittel.

Die *Operation* wird um die Art der Bearbeitung (*Roughing, Finishing*), Bearbeitungsaufmaße (*Allowance*) und Informationen für An-, Abfahr- und Verbindungsbewegungen (*Start Point, Approach, Retract, Retract Plane*) ergänzt. Diese Informationen werden in der NC-Steuerung zur Verfahrwegberechnung, Parametrierung der Überwachung und der Bereitstellung angepasster Regelstrategien ausgewertet. Die Technologieparameter (*Technology*) werden wahlweise maschinenorientiert (*Spindle Speed, Feed Rate*), werkzeugorientiert (*Tool, Feed Rate per Tooth*) oder prozessorientiert (*Cut Speed, Surface Roughness*) programmiert und können um Vorgaben zur adaptiven Regelung (*Adaptive Control*) ergänzt werden. Aufgrund des objektorientierten Aufbaus können die Definitionen erweitert oder durch ein abgeleitetes neues Datenelement an die anwenderspezifischen Anforderungen angepasst werden (Bild 5.8).

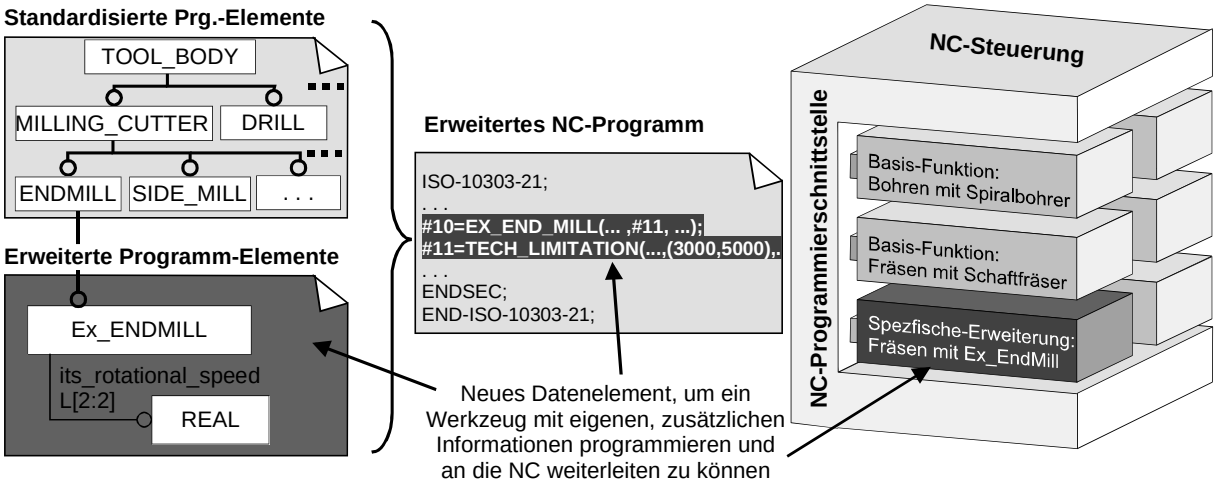


Bild 5.8: Paralleler Austausch standardisierter und neuer, spezialisierter Programmelemente
Common exchange of standardized as well as new, specialized program elements

5.2. Prozessverfolgung und -eingriffe über NC-Programmierobjekte

Process tracing and manipulation based on NC-programming objects

Die für die NC-Programmierschnittstelle entwickelten Objekte erlauben es, die zu programmierende Bearbeitung ausgehend von einer Arbeitsschrittfolge über die darin zu bearbeitenden Fertigungsmerkmale und anzuwendenden Operationen bis hin zu expliziten Verfahrwegen, Schaltanweisungen und Prozessparametern strukturiert zu kodieren. Dabei gliedern die Objekte den Bearbeitungsprozess in unterschiedliche Abstraktionsebenen. Diese Ebenen folgen dem Ablauf der NC-Programmierung. Sie spiegeln das aufgaben-, merkmals- und prozessorientierte Denken des Anwenders wider. In der Steuerung ermöglichen sie eine strukturierte Programmverarbeitung vom Arbeitsplan und der darin definierten Nullpunktverschiebung, über die Anwahl von Werkzeugen und Technologieparametern bis hin zu expliziten Verfahrwegen und Schaltanweisungen. Ausgehend von der umfangreichen Aufgaben- und Geometriebeschreibung können dem Werker der Bearbeitungsablauf dargestellt und situationsgerechte Eingriffsparameter bereitgestellt werden (Bild 5.9).

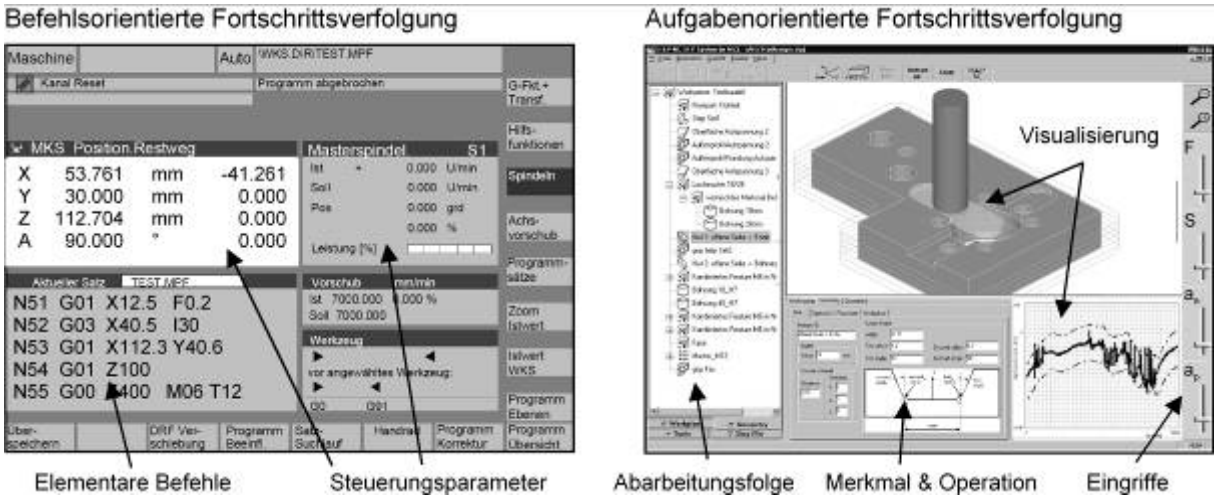


Bild 5.9: Aufgabenorientierte Programmanzeige, Fortschrittsverfolgung und Prozesseingriffe

Task oriented displaying of NC-Programs, progress and manipulatable parameters

5.2.1. Aufgaben- und prozessorientierte Fortschrittsverfolgung

Task and process oriented tracing of the machining process

Mit dem Ziel, die umfangreichen NC-Programmierobjekte für den Anwender sichtbar und nutzbar zu machen, werden sie als manipulierbare Datenstrukturen in die Benutzerschnittstelle und in die NC-Programmverarbeitung des Steuerungskonzepts integriert. Im Vergleich zu den elementaren Steuerungsbefehlen der DIN-Programmierung lassen sich so zusammenhängende Aufgaben, Geometrien und Parametersätze vollständig darstellen. Mit NC-Zyklen ist dies nur bedingt möglich. Sie sind herstellerspezifische Programmierhilfen, deren Dateninhalt auf die steuerungstechnische Abarbeitung eines Arbeitsschritts beschränkt ist. Ihnen fehlen beispielsweise Datenelemente, wie Nullpunktverschiebungen, Werkzeugdefinitionen oder Strategien zum globalen An- und Abfahren. Diese Informationen müssen, soweit dies ohne eine Aufgabenbeschreibung möglich ist, aus den vorangehenden DIN-Codes rückwärtig interpretiert werden.

Die ersten greifbaren Objekte nach dem Öffnen des konzipierten NC-Programms sind der zentrale Arbeitsplan und die darin referenzierten Arbeitsschritte. Ausgehend von dieser Struktur lassen sich dem Werker die zu verwendenden Rohteile, deren Spannlagen, die zu rüstenden Werkzeuge sowie die geplante Bearbeitungsfolge in der Benutzeroberfläche anzeigen. Dadurch kann der Werker die Maschine auf der Basis des NC-Programms einrüsten und den Bearbeitungsablauf ohne zusätzliche Dokumente oder ein textbasiertes Durchgehen elementarer Programmcodes prüfen. Die Visualisierung der mitunter umfangreichen NC-Programminformationen übernehmen grafische Anwendungen in der Benutzungsoberfläche. Diese stellt entweder der Steuerungshersteller bereit oder sie werden anwenderspezifisch entwickelt und zusätzlich in die Oberfläche integriert.

Während der Bearbeitung unterstützen die NC-Programmierobjekte und ihre hierarchische Struktur den Werker beim schrittweisen Erproben, Optimieren und Verfolgen des NC-Programms. Da die Arbeitsschritte als zusammenhängende Einheit im NC-Programm verfügbar sind, lassen sie sich parallel zur Abarbeitung aufrufen, ihre Parameter anzeigen und das Bearbeitungsziel sowie der Abarbeitungsfortschritt grafisch simulieren. Der Werker erkennt unmittelbar das Ziel des anstehenden Arbeitsschritts, kann virtuell die nachfolgenden Schritte betrachten und so rechtzeitig in den Ablauf eingreifen (Bild 5.9 rechts).

Im Einzelsatzbetrieb ist es möglich, die Bearbeitung gezielt zwischen einzelnen Arbeitsschritten, vor oder nach einer Zustellbewegung und, wie von konventionellen NC-Steuerungen bekannt, nach jeder explizit programmierten Weg- und Schaltanweisung zu unterbrechen. Darüber hinaus bietet sich die aufgaben- und prozessorientierte Struktur an, um beispielsweise nach einer Bearbeitungsebene, nach einem Werkzeugwechsel oder vor einer Änderung der Prozessparameter definiert anzuhalten. Die verschiedenen, dem Bearbeitungsablauf oder technologisch zusammenhängenden Prozessschritten entsprechenden Unterbrechungsmomente lassen sich Dank der aufgaben- und prozessorientierten NC-Programmierschnittstelle im NC-Programm direkt wieder finden und anwählen. Indem die grafische Benutzeroberfläche die Programmelemente anzeigt und als Unterbrechungspunkte anbietet, kann sie der Werker komfortabel vor- oder während der Bearbeitung anwählen. Die geforderte Unterbrechung wird an die Steuerung gemeldet, die die Programmabarbeitung definiert unterbricht und das Erreichen des Programmelements an den Werker zurückmeldet.

5.2.2. Aufgaben- und prozessorientierte Bearbeitungseingriffe

Task and process oriented interaction with the machining process

Wie im vorhergehenden Kapitel dargestellt wurde, ist die Strukturierung des Bearbeitungsablaufs in Arbeitsschritte geeignet, um den Prozess an vordefinierten Stellen zu unterbrechen, schrittweise einzufahren und den Anwender bei der Verfolgung der Programmabarbeitung zu unterstützen. Derartige Unterbrechungen erlauben jedoch keine bearbeitungsparallelen Eingriffe. Sie stellen lediglich ein Pausieren in der steuerungsintern vorverarbeiteten, zwischengespeicherten und auszuführenden Abarbeitungsfolge dar. Korrekturen am NC-Programminhalt oder an dessen Umsetzung erfordern weiterhin das zeitintensive Vorgehen von Programmabbruch, -änderung und -neustart.

Diese Einschränkung lässt sich durch eine flexible, sich periodisch oder ereignisgesteuert aktualisierende NC-Programmverarbeitung aufheben. Dazu ist es erforderlich, dass das NC-Programm nicht nur einmalig eingelesen, interpretiert und in ausführbare Sollwerte umgesetzt wird. Vielmehr müssen die Algorithmen der NC-Steuerung das NC-Programm fortlaufend auf Änderungen prüfen und unter Berücksichtigung steuerungs- und prozesstechnischer Abhängigkeiten neu umsetzen. Zu diesen Abhängigkeiten gehören beispielsweise die zu erwartende Reaktionszeit des Gesamtsystems, Weg- und Leistungsbegrenzungen der Maschinenachsen oder technologische Beschränkungen, wie der Werkzeugverschleiß und die Werkzeugform. Bezogen auf die dem Anwender zur Verfügung stehenden Eingriffsmöglichkeiten resultiert daraus, dass diese mit abnehmendem zeitlichem Vorlauf zur aktuellen Ausführung eingeschränkt werden müssen. Ausgehend von der Zeit bis zur Abarbeitung werden nachfolgend zwei Szenarien unterschieden:

1. Eingriff in zukünftige Arbeitsschritte und Programmbefehle vor deren Ausführung
2. Eingriffe in verknüpfte und aktuell auszuführende Arbeitsschritte und Befehle

a) Eingriff in zukünftige Arbeitsschritte und Programmbefehle

Das NC-Programm wird von der NC-Steuerung mit Vorlauf zur eigentlichen Ausführung eingelesen und verarbeitet. Der Vorlauf gleicht Lese- und Rechenzeiten der NC-Steuerung aus, wird zum rechtzeitigen Erkennen von Programmierfehlern oder möglichen Kollisionen genutzt und ermöglicht die Berechnung befehlsübergreifender, harmonischer Achsbewegungen und Schaltabläufe. Diese Abhängigkeiten müssen beim Zugriff auf die NC-Programmvorgaben und deren Umsetzung berücksichtigt werden. Eine geeignete Möglichkeit Vorgaben zu ändern, ohne dabei die gepufferte und in Ausführung befindlichen Sollwerte berücksichtigen zu müssen, bietet sich immer dann an, wenn das Werkzeug sich nicht im Bearbeitungseingriff befindet und Maschinenfunktionen vollständig abgeschlossen wurden. Dies trifft beispielsweise auf Programmvorgaben zu, die auf Rückzugs- und Eilgangbewegungen, einen Werk-

zeugwechsel oder eine zu quittierende Programmunterbrechung folgen. In diesen Fällen steht den Rechenroutinen der NC-Steuerung ausreichend Zeit zur Verfügung, um die geänderten Vorgaben zu interpretieren, ihre Gültigkeit zu prüfen und sie rechtzeitig in Sollwerte umzusetzen. Selbst wenn eine Umsetzung besonders rechenzeitintensiv wäre, die Vorgaben Fehler enthielten oder sich aufgrund von Achsbeschränkungen o.ä. nicht ausführen ließen, kann die Steuerung aus einer solchen Situation heraus die Abarbeitung in letzter Instanz verlangsamen oder unterbrechen. Solange sich das Werkzeug nicht im Eingriff befindet, haben die Reduzierung der Vorschubgeschwindigkeiten oder Pausen keine negativen Auswirkungen auf die Werkstückqualität. Wenn zudem die Steuerung sicherstellt, dass alle synchron ablaufenden Prozesse, wie Wechselsysteme, ebenfalls definiert angehalten werden, sind Gefährdungen des Werkers und der Maschine ausgeschlossen.

Zur Umsetzung dieses Ansatzes bietet sich ein Konzept an, in dem steuerungsintern ein verarbeitungs- sowie prozessorientiertes Modell der im NC-Programm kodierten Bearbeitungsaufgaben nachgeführt wird. Zu den einzelnen Arbeitsschritten werden dazu Ablaufmuster hinterlegt, die steuerungstechnisch eindeutig in Bezug auf zulässige Prozesseingriffe und Strategien zur sicheren Programmunterbrechung ausgewertet werden können. Indem die NC-Steuerung die aktuell ausgeführten als auch die anstehenden Aufgaben verwaltet, kann sie die Eingriffsmöglichkeiten des Werkers oder die eines externen Regelungssystems koordinieren. Entsprechend dem Abarbeitungsfortschritt sperrt sie Programmelemente vor externen Eingriffen oder gibt die noch ausstehenden, nicht von der laufenden Bearbeitungsausführung abhängenden Programmvorgaben zur Manipulation frei. Je nach softwaretechnischer Umsetzung erfolgen die Eingriffe entweder über Änderungen des NC-Programms oder über steuerungsseitig angebotene Direktschnittstellen auf die intern gespeicherten und vorverarbeiteten Programmvorgaben.

Die Benutzerschnittstelle reagiert auf die von der Steuerung für Eingriffe freigegebenen Arbeitsschritte, indem sie dem Werker kontextabhängige Interaktionsmechanismen anbietet. Hierzu eignen sich beispielsweise interaktive Grafiken, Auswahllisten, Schiebe- und Drehregler. Diese werden von den Anwendungen der Benutzerschnittstelle in Abhängigkeit von der aktuellen Operation und dem zu bearbeitenden Merkmal mit Parametern und Auswahloptionen vorbelegt. Somit können die Eingriffe schnell und mit möglichst wenigen Fehleingaben erfolgen. Die erforderlichen Sollvorgaben werden aus den Parametern des NC-Programms und den in der Steuerung implementierten Regelfunktionen abgeleitet. Die Ist-Prozessdaten werden aus den Schnittstellen der NC-Steuerung oder externer Überwachungssysteme ausgelesen.

Dieses Konzept erlaubt es, Nullpunktverschiebungen (*Setup*), die Arbeitsschrittfolge (*Workplan, Workingstep*), Werkzeuge (*Machining Tool*), An- und Abfahrbewegungen (*Approach, Retract*), Strategien (*Machining Strategy*), Schnittparameter (*Technology*) etc. ohne eine Programmunterbrechung vorausschauend in der Oberfläche zu korrigieren.

b) Eingriffe in verknüpfte, aktuell auszuführende Programmvorgaben

Mit fortschreitendem Grad der Programmumsetzung und dem Zugriff der SPS und Antriebe auf die dabei generierten Sollwerte, nehmen die Eingriffsmöglichkeiten in die NC-Programmvorgaben und deren Umsetzung ab. Beispielsweise kann aufgrund der im Antriebsregler gepufferten Sollwerte und der Trägheit des Antriebsstrangs das Profil einer Achsbewegung nicht beliebig geändert werden. Zusätzlich zu diesen mechanischen und steuerungstechnischen Einschränkungen müssen technologische Vorgaben berücksichtigt werden. So könnte zwar die Zustellung eines Schaftfräsers durch Verrechnung von Offsets kontinuierlich manipuliert werden, jedoch würden Rampen auf der Bauteiloberfläche stehen bleiben, die in der nachfolgenden Bearbeitungsebene zu wechselnden Zerspanungsquerschnitten führen und damit unerwünscht schwankende Prozesskräfte verursachen werden.

Damit diese vom Abarbeitungsfortschritt und der Bearbeitungssituation abhängigen Beschränkungen der Eingriffsoptionen für den Anwender und ein Regelsystem erkennbar sind, ist eine enge Abstimmung zwischen der NC-Steuerung und den extern eingreifenden Systemen notwendig. Hierzu werden in der Benutzerschnittstelle und der NC-Steuerung operations- und merkmalsspezifische Bearbeitungsmodelle implementiert. Innerhalb dieser Modelle sind die variablen Parameter und ihr Gültigkeitszeitraum eindeutig definiert. Damit Wechselwirkungen nicht an anderer Stelle zu Grenzwertverletzungen führen, werden die Parameter innerhalb der interagierenden Systeme und mit den im NC-Programm vorgegebenen Gültigkeitsbereichen abgeglichen. Anderenfalls könnten Gewindebohrer beim Zurückziehen abgebrochen, Schneiden tiefer als ihre wirksame Kantenlänge zugestellt oder Vorschübe so stark erhöht, dass kinematische Rauhigkeitsforderungen überschritten werden.

5.3. Modular erweiterbare Software-Architektur

Modular, scalable software architecture

Aus der Sicht des Anwenders besteht das Gesamtkonzept aus drei Subsystemen, der **Benutzerschnittstelle**, der **NC-Steuerung** und zusätzlichen **Überwachungs- und Regelsystemen**. Alle Funktionen zur Unterstützung des Anwenders, zur autonomen Auflösung der Bearbeitungsaufgabe, zur Prozessüberwachung oder zur Prozessregelung werden in diese drei Systeme integriert (vgl. Bild 5.2).

Aufgrund der vielfältigen Technologien, Maschinenfunktionen und der Forderung, spezifisches Prozesswissen in das Gesamtsystem integrieren zu können, bietet sich zur Umsetzung der dargestellten Eingriffsmethoden kein in sich abgeschlossenes, vom Steuerungshersteller allein bereitzustellendes System an. Stattdessen ist ein Grundsystem vorzusehen, in das die prozess- und anwenderspezifischen Algorithmen zur Abarbeitung, Verfolgung und Manipulation der Arbeitsschritte modular integriert werden können. Um Basisfunktionen nicht redundant entwickeln zu müssen, werden in den Grundsystemen der Benutzerschnittstelle und der NC-Steuerung alle steuerungsspezifischen Grundfunktionen und Sicherheitsroutinen vom Steuerungshersteller bereitgestellt. Hierzu zählen beispielsweise die Verriegelung von Funktionen in Abhängigkeit vom aktuellen Steuerungsmodus, der Zugriff auf das NC-Programm, die Umsetzung einfacher Programmvorgaben, die Geometriekette zur Berechnung der Achsbewegungen oder die Kopplung mit der SPS und den Antrieben. Darauf aufbauend erweitert der Maschinenhersteller das System um die für seine spezifischen Maschinenfunktionen und -kinematiken erforderlichen Umsetzungs- und Eingriffsroutinen. Dieser Ansatz ist mit der heutigen Integration von NC-Zyklen vergleichbar. Jedoch umfassen die hinzuzufügenden Module nicht nur Abbildungsroutinen, die einen Zyklus in elementare Steuerungsbefehle auflösen, sondern sie liefern zusätzlich Schnittstellen und Algorithmen, mit denen sich die Programmvorgaben und deren Umsetzung bearbeitungsparallel manipulieren lassen.

Dem Entwickler bietet der modulare Systemaufbau verschiedene Ebenen zur Integration seiner spezifischen Funktionen an. Die Struktur der Ebenen leitet sich zum einen aus den konventionellen Funktionsbereichen der NC-Steuerung und ihren Schnittstellen ab. Darüber hinaus werden die im NC-Programm enthaltenen Arbeitsschritte mit ihren Aufgaben-, Geometrie- und Operationsbeschreibungen als Schnittstellen zur Integration eigener Algorithmen angeboten. Bei der Entwicklung eigener Funktionen kann der Entwickler auf die vom Grundsystem bereitgestellten Routinen zurückgreifen und diese wiederverwenden. Alternativ kann er sie durch eigene, optimierte Implementierungen ersetzen. In Abhängigkeit von der gewählten Programmiersprache und dem Umfang seiner Erweiterung muss der Entwickler seine Softwaremodule zusammen mit dem Grundsystem neu kompilieren oder er kann sie als Bibliotheken dynamisch hinzufügen.

Der Einsatz von DLLs (Dynamik Link Libraries) bietet eine besonders große Flexibilität, da sie zur Laufzeit durch den Anwender konfiguriert, dynamisch in den Arbeitsspeicher geladen und aktiviert werden können. Dazu muss der Anwender die Erweiterungsmodule lediglich auf den Steuerungsrechner kopieren und sie in die beschriebenen Konfigurationsdateien der Subsysteme eintragen. Die Aktivierung erfolgt automatisch mit dem Systemstart oder dem Laden des nächsten NC-Programms. Dieser flexible Ansatz ist bereits bei Office Anwendungen, Systemupdates oder Treibern (vgl. Druckerinstallation) stark verbreitet.

5.3.1. Kommunikationsschnittstelle auf der Basis des NC-Programms

Communication interface based on the NC-program

Mit dem Ziel, auf bestehenden NC-Steuerungen aufsetzen zu können und potentielle Migrationswege aufzuzeigen, orientiert sich die Struktur des entwickelten Systemaufbaus an konventionellen NC-Funktionsbereichen und Schnittstellen. Dazu muss das Konzept auch die gängige soft- sowie hardwaretechnische Trennung von Benutzeroberfläche, NC-Steuerung und den darin implementierten Subsystemen ermöglichen.

In konventionellen Steuerungskonzepten wird das NC-Programm als Datei zwischen den Teilsystemen ausgetauscht. Jede der Applikationen liest die Programmvorgaben individuell ein und verhindert durch das Setzen von Schreib- und Leserechten Zugriffskonflikte (Bild 5.10 oben). Diesem Ansatz entspricht unter anderem das Vorgehen der Siemens *Sinumerik 840D* oder der Bosch Rexroth *MTX*. Hier lesen sowohl der NC-Editor, eine zusätzliche interaktive Programmierhilfe als auch der NC-Kern die NC-Programmdatei ein. Ein Programmstart ist dagegen nur möglich, wenn diese Anwendungen das Programm freigeben und sichergestellt wird, dass die Steuerung alleinigen Zugriff auf die Datei erhält. Bei den parallel zur Bearbeitung angezeigten Programmzeilen handelt es sich um reine Textinformationen. Diese werden von der NC-Steuerung in Abhängigkeit vom Abarbeitungsfortschritt zeilenweise an die Oberfläche geschickt. Nachteilig an diesem Ansatz ist, dass nach dem NC-Start nur noch lesend auf das Programm zugegriffen werden kann und manipulierende Eingriffe nur nach einem Programmabbruch, d.h. der Schreibfreigabe des Programms möglich sind.

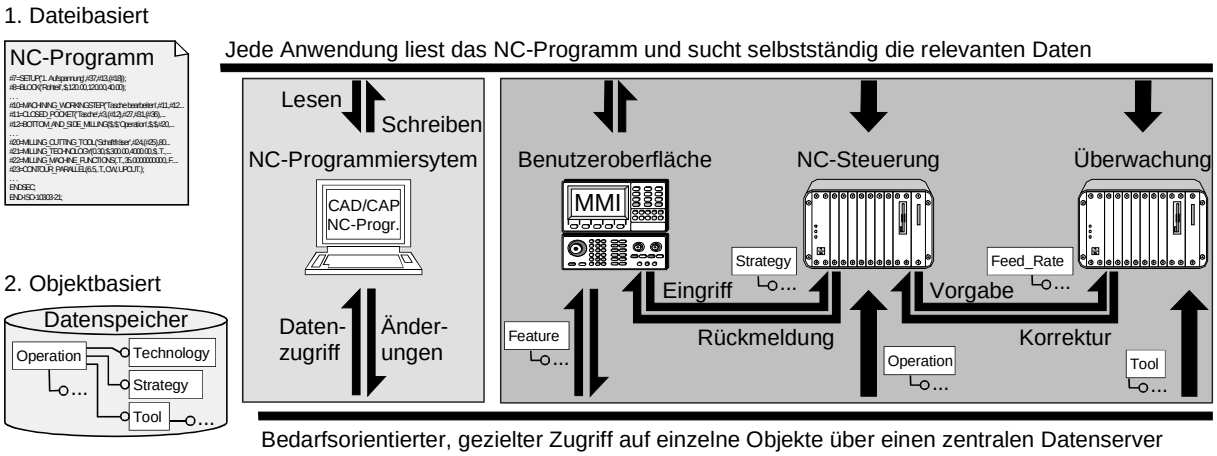


Bild 5.10: Systemkopplung auf der Basis der NC-Programmierschnittstelle

Data exchange and linkage based on the NC-programming interface

Um dennoch die geforderten Programmeingriffe zuzulassen, bietet sich ein Konzept mit einer zentralen NC-Programmverwaltung und einem Mechanismus zum bedarfsorientierten Austausch der programmierten und nach Eingriffen manipulierten NC-Programmelemente an (Bild 5.10 unten). Die Grundlage für diese Lösung leitet sich aus der objektorientierten Da-

tenstruktur der NC-Programmierschnittstelle ab. Demnach werden die einzelnen Programmelemente als Datenpakete über breitbandige Datenschnittstellen, wie z.B. das Ethernet, oder über einen gemeinsamen Arbeitsspeicher von den Teilsystemen abgefragt oder gezielt als Fortschrittmeldung und Korrekturen ausgetauscht. Dieser Ansatz ist heute bei zentral verwalteten Datenbanken mit verteilten Zugriffsystemen, den sogenannten Clients üblich.

Die Zugriffsrechte können über den Datenserver eingeschränkt werden. So ist es möglich in der Serienfertigung Änderungen an zertifizierten NC-Programmen auszuschließen. Ferner könnte die Datenverwaltung des Servers gezielt Programmelemente, wie die Operationen und Feature sperren und gleichzeitig die an der Maschine anzupassenden Elemente, wie Rohteilabmaße, Setup und Werkzeuge freigeben. Der Server selber kann entweder zentral sein und verwalten, ob ein Programm offline im CAM geändert wird oder online durch die NC abgearbeitet wird. Auch ist es denkbar in Analogie zu DNC-Steuerungen dezentrale Server auf dem jeweiligen Steuerungsrechner einzusetzen. Somit hat dann die Steuerung unabhängig von einem Netzwerk direkten Zugriff auf das NC-Programm. Dieser Ansatz wurde hier realisiert.

Die Teilsysteme (WOP, NC, Überwachung, Regelung) fragen die NC-Programmelemente in Abhängigkeit von ihrem Datenbedarf ab oder sie werden aktiv und direkt über Änderungen informiert. Zur Abfrage nutzen sie die referenzierende Struktur der NC-Programmierschnittstelle und greifen beginnend beim „*Project*“ auf den „*Workplan*“, die darin gelisteten „*Workingsteps*“, deren „*Operationen*“ und „*Feature*“ bis auf einzelne „*Technology*“-Vorgaben und „*Toolpaths*“ zu. Die für eine Anwendung irrelevanten Zusatzinformationen werden dabei ignoriert. Beispielsweise wird ein auf diesem Ansatz basierendes Überwachungssystem nur die relevanten Überwachungsparameter, wie die Werkzeug-, Maschinen- und Prozesskenndaten abrufen. Unbekannte Informationen werden als Warnung oder Fehlermeldung zur Anzeige gebracht oder im Vorfeld des jeweiligen Arbeits- oder Prozessschritts zur Programmunterbrechung und Quittierung durch den Anwender genutzt.

Der aktive, direkte Datenaustausch erfolgt nach einem Eingriff in die NC-Programmvorgaben. Dazu registriert die zentrale Datenverwaltung die Änderung an einem Datenelement und meldet diese weiter. Ferner bieten sich die Objekte der NC-Programmierschnittstelle an, um direkt zwischen den Teilsystemen ausgetauscht zu werden. Dieser Mechanismus ist dann sinnvoll, wenn nicht das NC-Programm geändert werden soll. Potentielle Anwendungen sind das Erstellen und Überprüfen von alternativen Arbeitsschritten, bevor diese in das NC-Programm übertragen und damit systemweit aktiv werden. Ferner sendet die NC-Steuerung vor dem Start eines neuen Arbeitsschritts die anstehenden Operationsdaten an die angeschlossenen Überwachungssysteme. Diese reagieren, indem sie ihre Eingriffsgrenzen und Überwachungsstrategien gezielt auf der Basis der gemeldeten und davon ausgehend referenzierten Datenelemente anpassen, ohne dass sie das gesamte NC-Programm lesen und interpretieren müssten. In der Gegenrichtung nutzen die Benutzerschnittstelle oder Regelsysteme die Datenstrukturen, um Änderungen aufgaben- und prozessorientiert zu kapseln und um diese zur Prozesskorrektur an die NC-Steuerung weiterzuleiten.

5.3.2. Durchgängige Systemerweiterung über die NC-Programmierschnittstelle

Consistent system extension based on the NC-programming interface

Wie dargestellt, werden über das Datenmodell der NC-Programmierschnittelle die geplanten Bearbeitungsaufgaben und alle Zusatzinformationen über den Prozessverlauf sowie mögliche Prozessalternativen in einem NC-Programm bereitgestellt. Diese Struktur und das darin abgebildete Prozessverständnis des Anwenders werden in die Datenstruktur der Benutzerschnittstelle und der NC-Steuerung übernommen. Somit bleibt der Kontext zwischen der Bearbeitungsaufgabe, der zu erzeugenden Geometrie, den anzuwendenden Strategien und Technologieparametern sowie den auszuführenden Schaltfunktionen und Verfahrwegen erhalten.

Das Konzept kapselt das Datenmodell, d.h. die NC-Programmierschnittstelle, in Form einer aufeinander verweisenden Objektstruktur und stellt diese als Bibliothek bereit. Die Bibliothek enthält neben den Zugriffsfunktionen zum Lesen und Schreiben des NC-Programms auch das Klassenmodell, um den Programminhalt innerhalb der verschiedenen Benutzer- und Steuerungsapplikationen zu verwalten (vgl. Kapitel 5.1.1).

Die spezialisierten Module der Benutzerschnittstelle und der NC-Steuerung binden die Bibliothek der NC-Programmierschnittstelle ein und greifen über ihre Schreib- und Lesefunktionen auf den Programminhalt zu. Sie sind dadurch unabhängig von dem physikalischen Programmformat, d.h. sie müssen dieses nicht selber analysieren und auswerten. Die Anpassung zwischen Dateiformat und internem Steuerungsformat übernehmen der Parser (Lesen) und der Formater (Schreiben) des NC-Programmierschnittstellenmoduls. Das Konzept kann flexibel an die zukünftigen physikalischen Speicherformate angepasst werden, wie beispielsweise einem DIN ähnlichen Klartext, objektorientierten Strukturen nach STEP ISO 10303-21 oder XML. Auch die Anbindung an Datenbanken ist denkbar. Dazu müssen nur die Parser- und Formaterfunktionen der Bibliothek durch die datenbankspezifischen Zugriffsmethoden ausgetauscht werden. Die Klassenstrukturen zum Speichern der Programmvorgaben sowie die Funktionen zu deren Umsetzung können unverändert beibehalten werden.

Darüber hinaus werden durch die Kapselung die Programminformationen konsequent von ihrer Interpretation, d.h. dem Abbilden auf das spezifische Prozessmodell und der funktionalen Umsetzung in den Applikationen getrennt. Erst in den Softwareschnittstellen der Benutzeroberfläche oder der NC-Steuerung werden die Programmelemente den spezialisierten Verarbeitungsalgorithmen zugewiesen, d.h. interpretiert, Bild 5.11. Eine Konfigurationsdatei erlaubt dem Anwender oder dem Entwickler flexibel eigene, spezialisierte Algorithmen in die Systeme zu integrieren und diese gezielt für die Elemente des NC-Programms aufzurufen. Nicht erkannte Elemente werden an Standardfunktionen (Default) weitergeleitet, die Warnmeldungen ausgeben oder den Programminhalt mit Hilfe von Basisalgorithmen umsetzen.

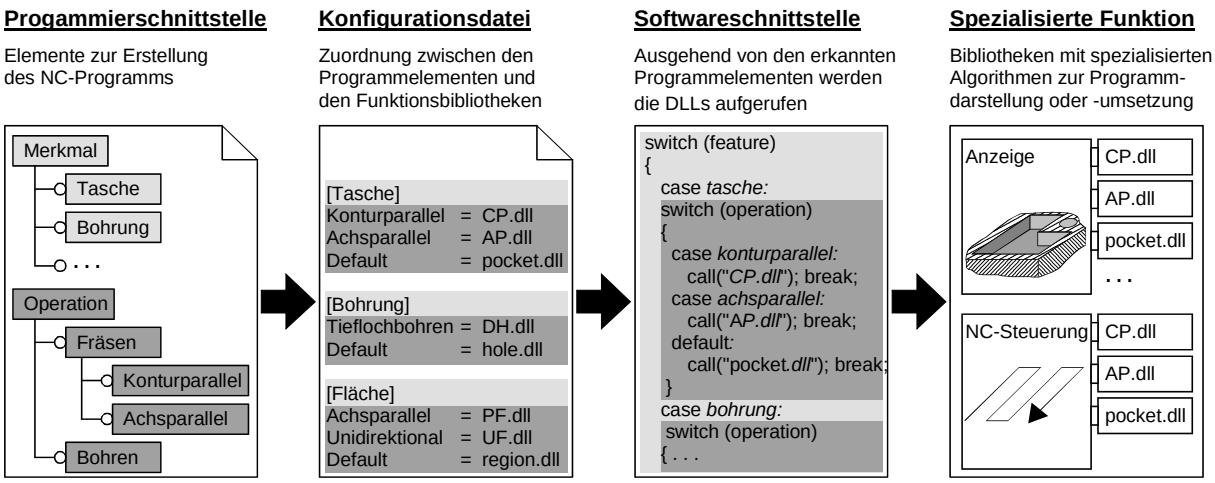


Bild 5.11: Flexible Funktionszuordnung zu den Elementen der NC-Programmierschnittstelle
Flexible linkage between functions and elements of the NC-programming interface

5.3.3. Aufgaben- und merkmalsorientierte Erweiterung der Funktionsmodule

Task and feature oriented extension of the software functions

Aufbauend auf der von der NC-Programmierschnittstelle vorgegebenen Struktur werden in den Teilsystemen Schnittstellen zur Integration eigener Umsetzungsfunktionen vorgesehen. Dazu unterscheidet die zentrale Plattform in Abhängigkeit vom Datentyp u.a. zwischen dem

Ablauf (*Workplan, NC-Function, Workingstep*), der Geometrie (*Region, Feature*), den Operationen und den Ressourcen (*Tools, …*). Ausgehend von dieser Unterscheidung übernimmt der Systementwickler die Struktur des Datenmodells in seine Implementierung und weist ihr seine Funktionen zur weiteren Umsetzung, Darstellung und Ausführung zu. Für den Anwender ergeben sich daraus eindeutig definierte Einsprungebenen. Er kann diese entsprechend seinen Anforderungen aus dem Basissystem der Benutzungsoberfläche, bzw. der Steuerung übernehmen oder durch individuelle Erweiterungen ersetzen. Gemäß den Anforderungen lässt sich das Gesamtsystem modular aus bestehenden und spezialisierten Funktionen konfigurieren und erweitern (vgl. Bild 5.8 und Bild 5.11).

Um die Sicherheit des Systems zu gewährleisten und undefinierte Zustände zu vermeiden, werden Routinefunktionen entwickelt, die eingreifen, wenn ein Datenelement nicht erkannt wird oder keine expliziten Funktionen hinterlegt sind. Diese Default-Funktionen sind vom Steuerungshersteller bereitzustellen. Sie weisen den Anwender zum einen bei der Anwahl des NC-Programms auf die fehlenden Funktionalitäten hin und zum anderen brechen sie die Bearbeitung in sicheren Zuständen ab, wie beispielsweise vor dem Programmstart, in einer definierten Werkzeugwechselposition oder bevor das Werkzeug in Eingriff kommt. Dieser Ansatz stellt sicher, dass der Zustand des Gesamtsystems für den Anwender jederzeit nachvollziehbar ist und er in Abhängigkeit von der steuerungsseitig eingestellten Zugriffsberechtigung zur Laufzeit die Umsetzungs-, Darstellungs- und Ausführungsroutinen dynamisch nachladen und konfigurieren kann.

Die typischen Module des Konzepts und ihre Funktionsweise werden systemspezifisch in Kapitel 6 für die **NC-Steuerung** und in Kapitel 7 für die **Benutzerschnittstelle** detailliert.

6. Steuerungskern zur adaptiven Programmverarbeitung

NC-kernel for adaptive processing of the NC-program

Im Stand der Technik wurde dargestellt, dass in konventionellen NC-Steuerungen einfache Befehlsfolgen unidirektional in steuerungsinterne Formate konvertiert und dann sequentiell abgearbeitet werden. Zur Laufzeit kann der Anwender den Inhalt oder den Ablauf der programmierten Befehle nicht manipulieren. Er hat nur noch Zugriff auf die Sollwertausführung, indem er z.B. die Vorschubgeschwindigkeit und die Spindeldrehzahl über Overrideschalter manipuliert. Diese Lösung genügt nicht den in Kapitel 0 aufgestellten Anforderungen nach aufgaben- und prozessorientierten Eingriffsmethoden in den laufenden Bearbeitungsprozess.

Ausgehend von dem in Kapitel 5 entworfenen Gesamtkonzept einer umfassenden Aufgabenbeschreibung im NC-Programm und der Prozessbeeinflussung über das NC-Programm, wird in diesem Kapitel das hierzu funktionell und ablauforganisatorisch erweiterte Steuerungskonzept detailliert. Darin werden die Vorgaben des NC-Programms in steuerungsinterne Arbeitsschritte gegliedert und bearbeitungsparallel zur Verfolgung sowie zur Manipulation des Bearbeitungsprozesses zur Verfügung gestellt. Die zu entwickelnden Teilfunktionen und Kommunikationsmechanismen der NC-Steuerung stellen sicher, dass Eingriffe in die NC-Programmvorgaben übernommen werden und die laufende Bearbeitung synchron adaptiert wird. Offene Schnittstellen ermöglichen die Integration eigener Programminterpretationsroutinen und Regelstrategien. Bild 6.1. gibt einen Überblick über das nachfolgend entwickelte Steuerungskonzept. Angelehnt an heutige Steuerungsarchitekturen, greift das Konzept in die grau hinterlegten Module ein, bzw. ersetzt den Interpreter und die Datenkommunikation.

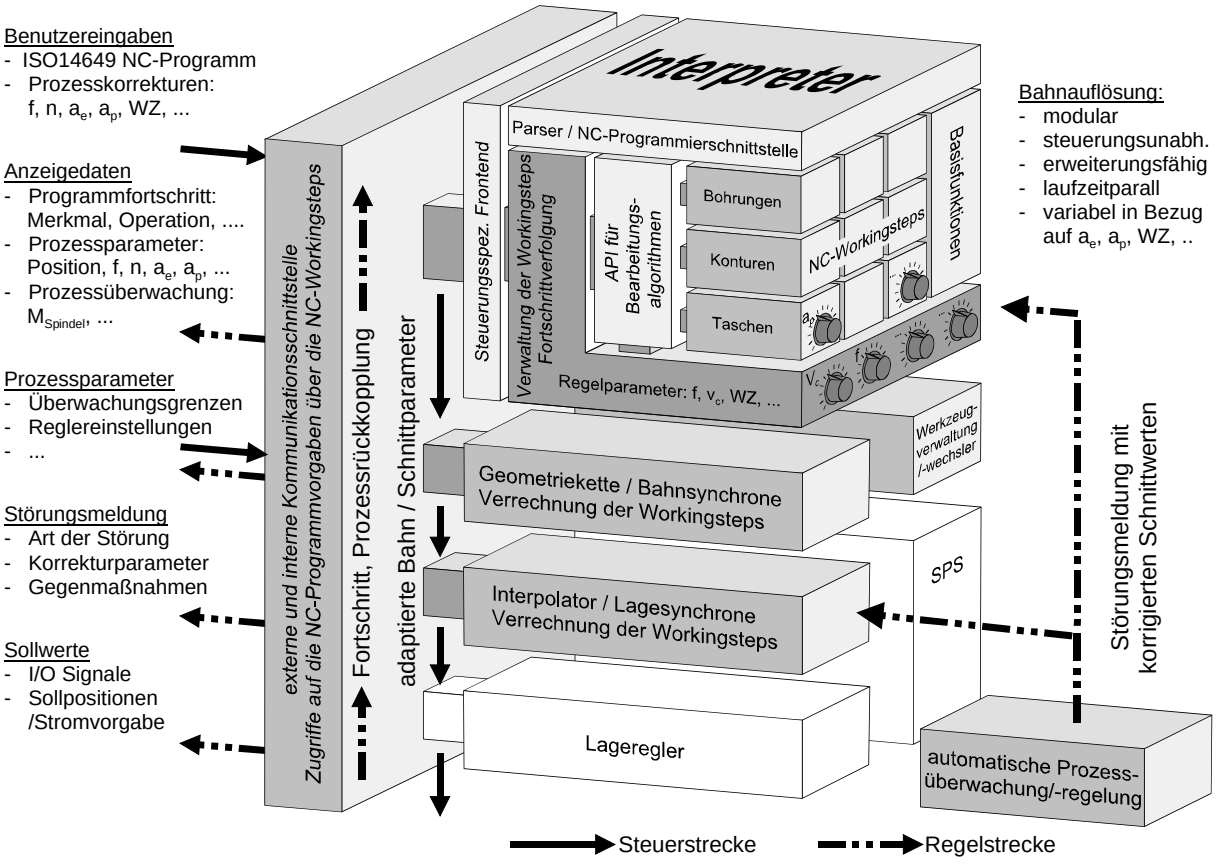


Bild 6.1: Konzept einer durchgängigen, modularen Verarbeitung von Bearbeitungsaufgaben

Concept of a consistent, modular processing of machining workingsteps

Das Konzept sieht vor, dass der NC-Interpreter die in Kapitel 5.1 entwickelten, zusammenhängenden Aufgabenbeschreibungen einliest und sie kontextorientiert auf ablauf- und steuerungsorientierte Aufgabenmodelle umsetzt. Innerhalb dieser Aufgaben, die im Folgenden als **NC-Workingsteps** (Kapitel 6.1) bezeichnet werden, bleiben die im NC-Programm kodierten Aufgabenbeschreibungen und Vorgaben durchgängig verfügbar und werden nach außen als Schnittstellen zur Fortschrittsverfolgung oder für Prozesseingriffe zur Verfügung gestellt.

Der **Interpreter** (Kapitel 6.2) löst die in den Arbeitsschritten des NC-Programms enthaltenen Vorgaben schrittweise in verkettete Listen elementarer, steuerungsspezifischer NC-Workingsteps auf, Bild 6.2. Hierzu werden steuerungs- und maschinenoptimierte Verfahrbewegungen, Anweisungen zum Schalten von Maschinenfunktionen und Synchronisationsmarken generiert. Mit dem Ziel, spezifisches Prozesswissen in der Programmauflösung einsetzen zu können, lassen sich die Umsetzungsroutinen des Interpreters miteinander kombinieren, erweitern oder durch spezialisierte Routinen eines Systementwicklers austauschen. Hierzu werden die NC-Workingsteps des Interpreters als eine offen erweiterbare Bibliothek realisiert.

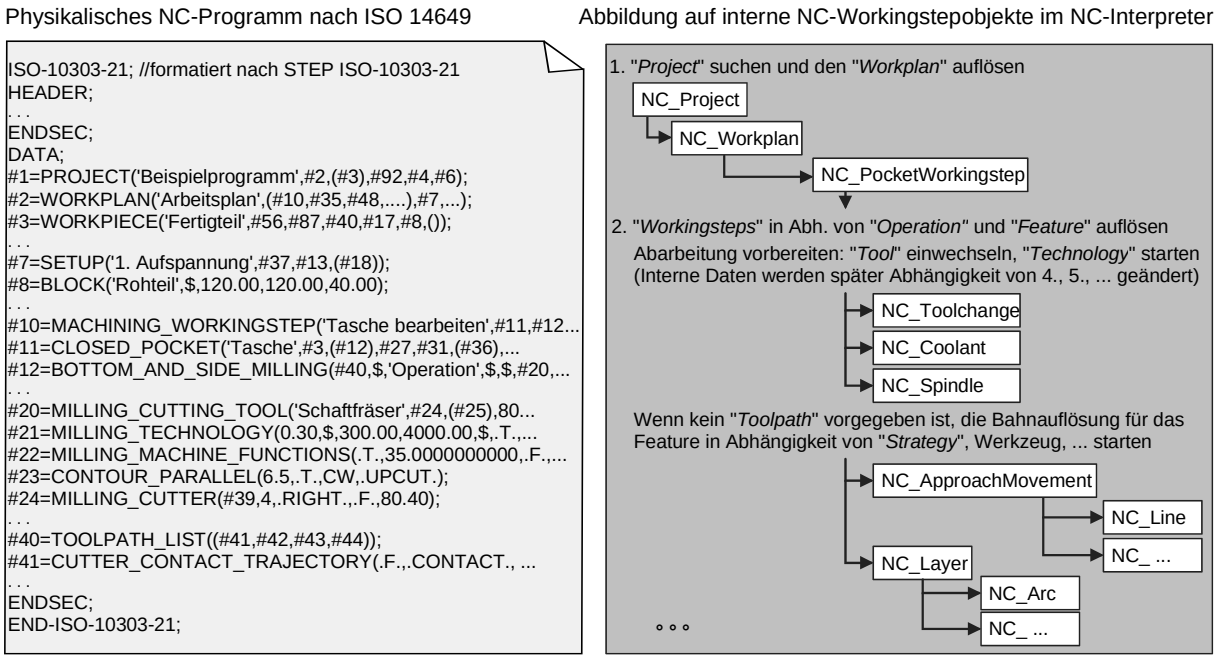


Bild 6.2: Abbildung des NC-Programms auf die interne Datenstruktur der NC-Steuerung
Mapping of the NC-program's content to the NC's internal data structure

Kommunikationsfunktionen stellen sicher, dass Parameter zwischen den Workingsteps ausgetauscht und extern bereitgestellt werden (Kapitel 6.3). Anhand der aktuellen Daten eines NC-Workingsteps lässt sich dessen Abarbeitungsfortschritt verfolgen und dem Anwender anzeigen. Externe Überwachungs- und Regelsysteme lesen die Workingsteps aus, um ihre Funktionen kontextorientiert an die Bearbeitung anzupassen. Eingriffe werden von den Workingsteps in Abhängigkeit von ihrem Abarbeitungsstand ausgewertet, untereinander abgeglichen und umgesetzt. Die finale Ausführung, Verzögerungen oder aufgetretene Fehler werden an die Kommunikationsschnittstellen zurückgemeldet.

Die **Geometriekette** und die **Ablaufsteuerung** (Kapitel 6.4) stellen sicher, dass die generierten elementaren Steuerungssignale den Kontext zu den übergeordneten NC-Workingsteps behalten. Dazu werden die generierten Bahnkurven und Schaltbefehle als Sub-Workingsteps an die genannten NC-Workingsteps angehängt. Dadurch können die Steuerungsfunktionen, wie die Werkzeugkorrektur, Transformation und die Geschwindigkeitsführung, jederzeit den Bezug zu der ursprünglichen, im NC-Programm kodierten Bearbeitungsaufgabe herstellen und möglichst schnell auf Eingriffe und adaptierte Vorgaben reagieren.

Die die NC-Workingsteps generierenden Steuerungsmodule sowie die NC-Workingsteps selber generieren in Abhängigkeit von ihrer Prozesslogik vorausschauend **Eingriffsoptionen sowie alternative Abläufe, Parameter und Strategien** (Kapitel 6.5). Dem Anwender oder einer Regelung werden so gezielt Optionen zur Manipulation der laufenden Bearbeitung angeboten. Des Weiteren kann die Steuerung die verschiedenen Alternativen steuerungsintern vorverarbeiten, um im Fall eines Eingriffs schnell reagieren zu können. Somit wird es möglich, nahe an den Signalschnittstellen der Steuerung einzugreifen. Diese Lösung kompensiert die durch Rechen- und Kommunikationszeiten verursachten Totzeitelemente der NC-Steuerung, bis dass vom Interpreter neu umgesetzte NC-Workingsteps mit adaptierten Schaltsignalen und Verfahrwegen vorliegen.

6.1. NC-Workingsteps zur steuerungsinternen Programmverarbeitung

NC-workingsteps for NC internal processing of the NC-program data

Die elementaren, an den Funktionen der Maschine orientierten Weg- und Schaltbefehle der DIN 66025 entsprechen nicht dem aufgabenorientierten Prozessverständnis des Anwenders. Sie sind für ihn schwer nachzuvollziehen. Aufgrund der Ausführungsgeschwindigkeit kann er sie in der laufenden Bearbeitung nicht gezielt manipulieren. Einzelne Prozessparameter, wie die Bahngeschwindigkeit oder die Spindeldrehzahl, lassen sich daher nur satzübergreifend mithilfe der Override regeln. Ein gezielter, aufgaben- und technologieorientierter Eingriff in einzelne NC-Sätze ist zur Laufzeit nicht möglich.

Diese Einschränkungen werden konzeptionell aufgehoben, indem während der Programmverarbeitung die Vorgaben technologie- und aufgabenorientiert in größeren, zusammenhängenden Einheiten zusammengefasst werden. Dazu bildet die Steuerung die Informationen aus der entwickelten NC-Programmierschnittstelle in Form von Datenobjekten auf interne Arbeitsschritte ab. Diese als NC-Workingsteps bezeichneten Objekte enthalten die ursprünglichen Programminformationen sowie die innerhalb der Programminterpretation und der Geometrieverarbeitung generierten Steuerungsinformationen. Der Anwender oder eine Regelung können über die NC-Workingsteps auf größere, den Bearbeitungsablauf strukturierende Einheiten zugreifen. Dies erlaubt die Realisierung der geforderten aufgabenorientierten Fortschrittsverfolgung und Prozessmanipulation.

6.1.1. Verknüpfte, interaktive NC-Workingsteps

Linked, interactive NC-workingsteps

Die in Kapitel 5.1 konzipierte NC-Programmierschnittstelle stellt der NC-Steuerung eine umfassende, zusammenhängende Geometrie-, Aufgaben- und Operationsbeschreibung bereit. Sie liefert alle Informationen über die Reihenfolge der abzuarbeitenden Prozessschritte, die zu fertigenden Merkmale und die dazu anzuwendenden Strategien, Werkzeuge und Technologieparameter. In Abhängigkeit von dem in der Arbeitsvorbereitung gewählten Detaillierungsgrad und der Notwendigkeit den Bearbeitungsablauf exakt vorzubestimmen, können für jeden Arbeitsschritt zusätzlich die Sequenz der auszuführenden Schaltfunktionen und Verfahrwege kodiert werden.

Dieser vom Arbeitsplan ausgehende, hierarchische Zusammenhang zwischen organisatorischen und an der Ausführung orientierten Programmdaten wird in der Programmumsetzung der NC-Steuerung beibehalten und über Schnittstellen nach außen offen gelegt. Somit kann der Werker auf der Basis seines Prozessverständnisses als auch eine Regelung in den Bearbeitungsprozess eingreifen. Zu den Eingriffsebenen gehören beispielsweise die Arbeitsschrittreihenfolge, Strategien, einzusetzende Werkzeuge, Technologieparameter und, eine schnelle Regelung vorausgesetzt, die final auszuführenden Wegsegmente und Schaltanweisungen.

Die informationstechnische Grundlage des Konzepts sind in der NC-Steuerung generierte und schrittweise in der Steuerung zu Sollwerten weiterverarbeitete Datenobjekte. Sie werden je nach NC-Programmvorgabe als Objekt eines spezifischen Typen, einer sogenannten Klasse generiert. Die verschiedenen Klassen unterscheiden sich je nach Aufgabe durch ihre Attribute und Funktionen (z.B. *NC_PocketWorkingstep, NC_Toolchange*). Im Folgenden werden die Klassen und Objekte als **NC-Workingsteps** bezeichnet. Das Präfix „NC“ soll sie von den „*Workingsteps“* des STEP-NC-Programms unterscheiden. Im Gegensatz zu den Programm-*Workingsteps* bilden sie die steuerungsinterne Programmverarbeitung bis hin zu elementaren NC-Workingsteps zur Sollwert-Übertragung ab.

Ein ähnlicher Ansatz wurde bereits von *Brouër* eingesetzt, um die im NC-Programm der „Autonomen Produktionszelle“ (APZ) enthaltenen Workingsteps durch den NC-Interpreter in eine Abfolge steuerungsinterner Schalt- und Wegbefehle aufzulösen [BROU00]. Hierzu implementierte er eine Klassenstruktur, die die verschiedenen Programm-*Workingsteps* abbildete und in Sequenzen elementarerer Steuerungs-Workingsteps untergliederte. Das Ergebnis dieser Art der Programminterpretation sind verkettete Objektstrukturen aus sogenannten Sub-Workingsteps, deren unterste Ebene einzelnen Wegbefehlen entsprach, die an die Geometriekette weitergeleitet wurden und dort konventionell zu Sollwerten umgesetzt wurden.

In die Lösung einer ausführungsparallelen Programmadaption wird aus *Brouërs* Ansatz die hierarchische Programmauflösung in Sub-Workingsteps übernommen. Dieses wird jedoch zusätzlich bis zu den Sollwertschnittstellen beibehalten. Somit kann jederzeit vorwärts als auch rückwärts von den elementaren Weg- und Schalt- NC-Workingsteps auf deren übergeordnete Bearbeitungsaufgaben zugegriffen werden, Bild 6.3.

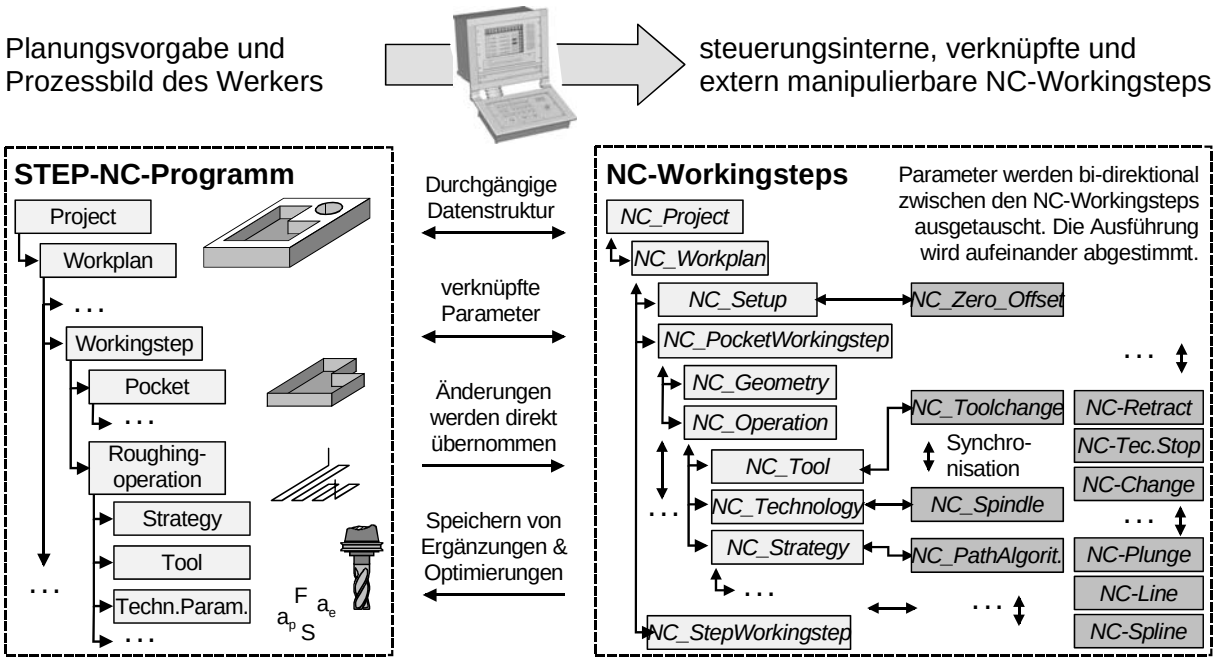


Bild 6.3: NC-Workingsteps als hierarchisch verkettete Objekte innerhalb der NC-Steuerung

NC-workingsteps as hierarchically linked objects inside the control’s memory

Eine zusätzliche Erweiterung gegenüber den Steuerungs-Workingsteps von *Brouër* ist die Interaktion der NC-Workingstepobjekte. Bei *Brouër* waren die Steuerungs-Workingsteps einmalig generierte, statische Datenelemente. Sie boten keine Schnittstellen an, über die sie extern oder in Abhängigkeit voneinander manipuliert werden konnten. Der Bearbeitungsprozess ließ sich nur regeln, indem eine übergeordnete Intelligenz im Interpreter die ausstehenden Schritte löschte, die Bearbeitung unterbrach und neue, adaptierte Schritte generierte.

Regeleingriffe waren nur vor dem Beginn eines neuen, direkt aus dem NC-Programm abgeleiteten Steuerungs-Workingsteps und nicht innerhalb der Ausführung einer Sub-Workingstepliste möglich. Bis zur Bereitstellung der neu zu generierenden Arbeitsschritte arbeitete der NC-Kern die gepufferten Listen ab oder die Bearbeitung musste abgebrochen werden. Im Fall einer Taschenbearbeitung bedeutete das, dass zuerst die aktuelle Zustellebene abgearbeitet wurde, bevor ein alternatives Werkzeug einwechselt werden konnte oder Technologieparameter, wie die Eingriffsbreite oder die Zustelltiefe sich adaptieren ließen.

Diese Einschränkung hebt die hier entwickelte Lösung auf, indem die Steuerungs-Workingsteps nicht als Sequenz statischer Datenstrukturen, sondern als untereinander verknüpfte Datenobjekte mit interagierenden Attributen- und Funktionen realisiert werden. Jedes dieser steuerungsinternen NC-Workingstepobjekte kann eigenständig die ihm entsprechenden Informationen des NC-Programms interpretieren und umsetzen. Es meldet seinen Verarbeitungsstatus und löst sich prozessorientiert schrittweise auf, bis hin zu elementaren Schaltanweisungen und Lagesollwerten. Auf Eingriffe reagieren die NC-Workingstepobjekte eigenständig, indem sie in Abhängigkeit von ihrem Abarbeitungsfortschritt ihre Algorithmen neu aufrufen, ihre Attribute adaptieren und die untergeordneten Sub-Workingstepobjekte sich adaptieren lassen oder diese neu generieren. Somit kann innerhalb eines Konturzuges jedes einzelne Bahnsegment auf die vorangehenden und nachfolgenden Segmente reagieren, Übergänge glätten, Technologieparameter übernehmen, adaptieren oder an nachfolgende NC-Workingsteps weiterleiten.

Eines der wesentlichen Funktionsmerkmale der hier neu eingeführten NC-Workingsteps ist, dass sie miteinander verknüpft sind und zusätzlich zu den Programmvorgaben ihre Attribute und ihr Verhalten aktiv aus vorangegangenen und nachfolgenden NC-Workingsteps ableiten. Auf diese Weise passen sie sich fortlaufend an die aktuell zurückgemeldete Bearbeitung an und reagieren kaskadiert auf externe Prozesseingriffe. Mithilfe der Verkettung werden die Änderungen an untergeordnete und nachfolgende NC-Workingsteps weitergeleitet, Bild 6.4.

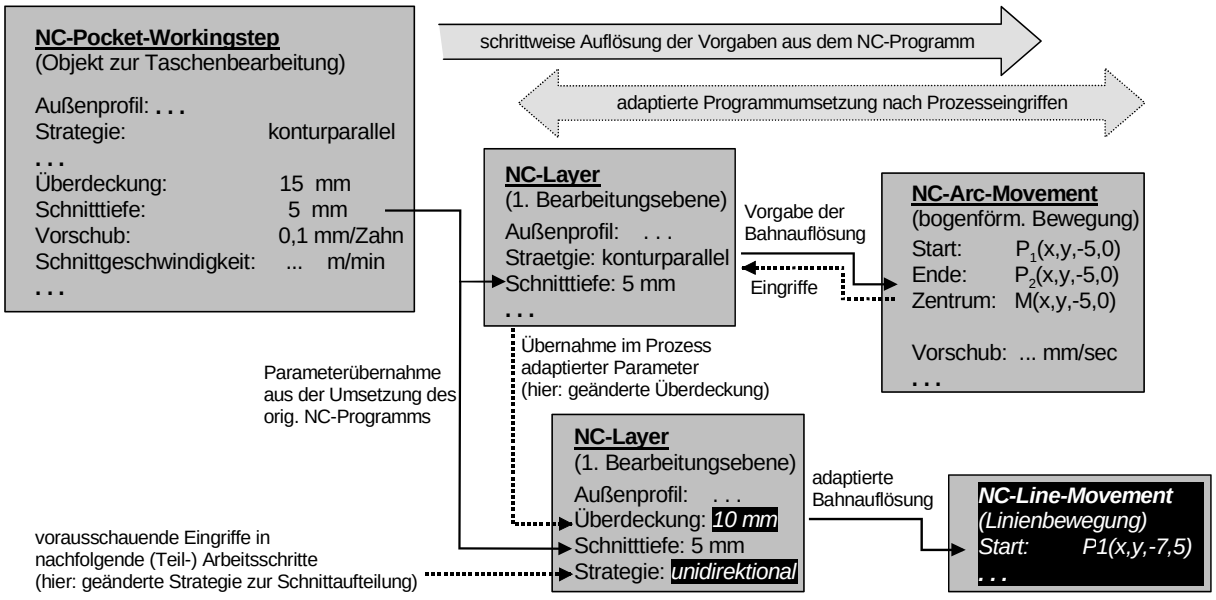


Bild 6.4: Schrittweise Auflösung einer Aufgabe in elementare, interagierende Teilaufgaben
Iterative break-down of a workingstep into simple, interactive sub-workingsteps

Dieser Mechanismus wird möglich, indem die einzelnen Workingsteps im Systemspeicher miteinander verknüpft werden. Somit können sie zum einen auf gemeinsame Attribute ohne redundante Kopien zugreifen und Umsetzungsergebnisse direkt untereinander austauschen.

Zu diesen Attributen zählen u.a. die Soll- und Ist-Daten der NC-Workingstepobjekte, Steuerungssignale oder die Eingaben über die Benutzerschnittstelle.

Ändert der Werker oder eine Prozessregelung einen Technologieparameter, so wird diese Änderung automatisch zwischen allen mit der Bearbeitungsaufgabe verknüpften NC-Workingstepobjekten ausgetauscht. Die Änderungen werden innerhalb der, das geänderte Merkmale verarbeitenden NC-Workingsteps erkannt, interpretiert und umgesetzt. Als Reaktion auf eine Änderung können sich nicht nur die Attribute und untergeordneten NC-Workingsteps eines Programm-*Workingsteps* ändern. Wird er überflüssig, so löscht ihn der übergeordnete NC-Workingstep und passt die Ausführungssequenz an. Gemäß der Aufgabenstellung erlaubt dieser Ansatz eine situationsgesteuerte, manipulierbare Programmumsetzung unter Zugriff auf die Merkmale des NC-Programms sowie auf die steuerungsinternen Abläufe, die in den NC-Workingsteps gekapselt sind.

6.1.2. Ablauf-, Aufgaben- und Funktionsbeschreibung

Description of workplans, workingsteps and NC-functions

Die zur Programmverarbeitung minimal erforderlichen, softwaretechnisch in der NC-Steuerung zu realisierenden NC-Workingsteps leiten sich aus der NC-Programmierschnittstelle, den technologie- und anwenderspezifischen Prozessmodellen sowie den nachfolgend durch die NC-Steuerung anzusteuernden Schalt- und Bewegungsfunktionen ab. Darüber hinaus sind spezialisierte NC-Workingsteps notwendig, über die sich der Ablauf logisch strukturieren lässt, die die Fortschrittsverfolgung und Interaktion unterstützen und mit denen sich die Programmvorgaben zu Sollwerten verrechnen lassen. Sie übernehmen Aufgaben, wie die Gruppierung mehrerer Abarbeitungsschritte, das Kapseln technologisch zusammenhängender Verfahrwege und Schaltabläufe, die Ausführung von Anwenderbenachrichtigungen und die Berechnung der Schnittaufteilung. Beispielsweise wird eine Taschenaufgabe nicht direkt in die zur Abarbeitung erforderlichen Verfahrwege und Schaltanweisungen aufgelöst. Stattdessen werden den elementaren NC-Workingsteps logische NC-Workingsteps übergeordnet, wie das Vorpositionieren, Zustellen und Abzeilen einzelner Bearbeitungsebenen oder das Zurückziehen des Werkzeugs („CNCApproach“, „CNCPlunge“, „CNCMillLayer“, …). Diese Workingsteps melden über Softwareschnittstellen ihre variablen Parameter und ihren Abarbeitungsstand nach außen. Zudem repräsentieren sie in sich geschlossene Bearbeitungsabläufe.

Dieser Ansatz einer Struktur aus logisch in sich abgeschlossenen und typisierten Arbeitsschritten unterstützt die vom Werker als auch einer Prozessüberwachung geforderte aufgabenorientierte Ablaufverfolgung. Sie gibt ausgehend von den NC-Workingsteps Einsprungfunktionen vor, zu denen gezielt in die Abarbeitung eingegriffen oder der Bearbeitungsablauf unterbrochen werden kann. Die hierarchische Workingstepstruktur ermöglicht unterschiedliche Zugriffsebenen ausgehend von den im NC-Programm definierten Arbeitsschritten bis hin zu elementaren Verfahrwegen, Bild 6.5. Der in der Aufgabenstellung geforderte, bearbeitungsparallele, aufgabenorientierte Eingriff ist somit bezogen auf das NC-Programm, einen Arbeitsschritt, untergeordnete Elemente, wie eine Zustell- oder Bearbeitungsbewegung, oder einzelne Wegsegmente möglich. Die verschiedenen Abstraktionsgrade bilden die Grundlage, um anwender- als auch situationsgerecht Eingriffsmethoden über die Steuerungsoberfläche bereitzustellen.

Kommt es zu Störungen, kann über die Softwareschnittstellen auf die Parameter der NC-Workingsteps zugegriffen werden und beispielsweise in der laufenden Fräsbearbeitung die Überdeckung, die Zustellung oder das Werkzeug geändert werden. In den ersten Fällen, d.h. einer Änderung des Zerspanungsquerschnitts, werden die den Verfahrweg beschreibenden NC-Workingsteps adaptiert oder neu berechnet. Für einen Werkzeugwechsel generiert die Steuerung zudem kollisionsfreie Abhebebewegungen zum Werkzeugwechselpunkt und stellt das neue Werkzeug gemäß einer ebenfalls extern parametrierbaren Zustellbewegung zu.

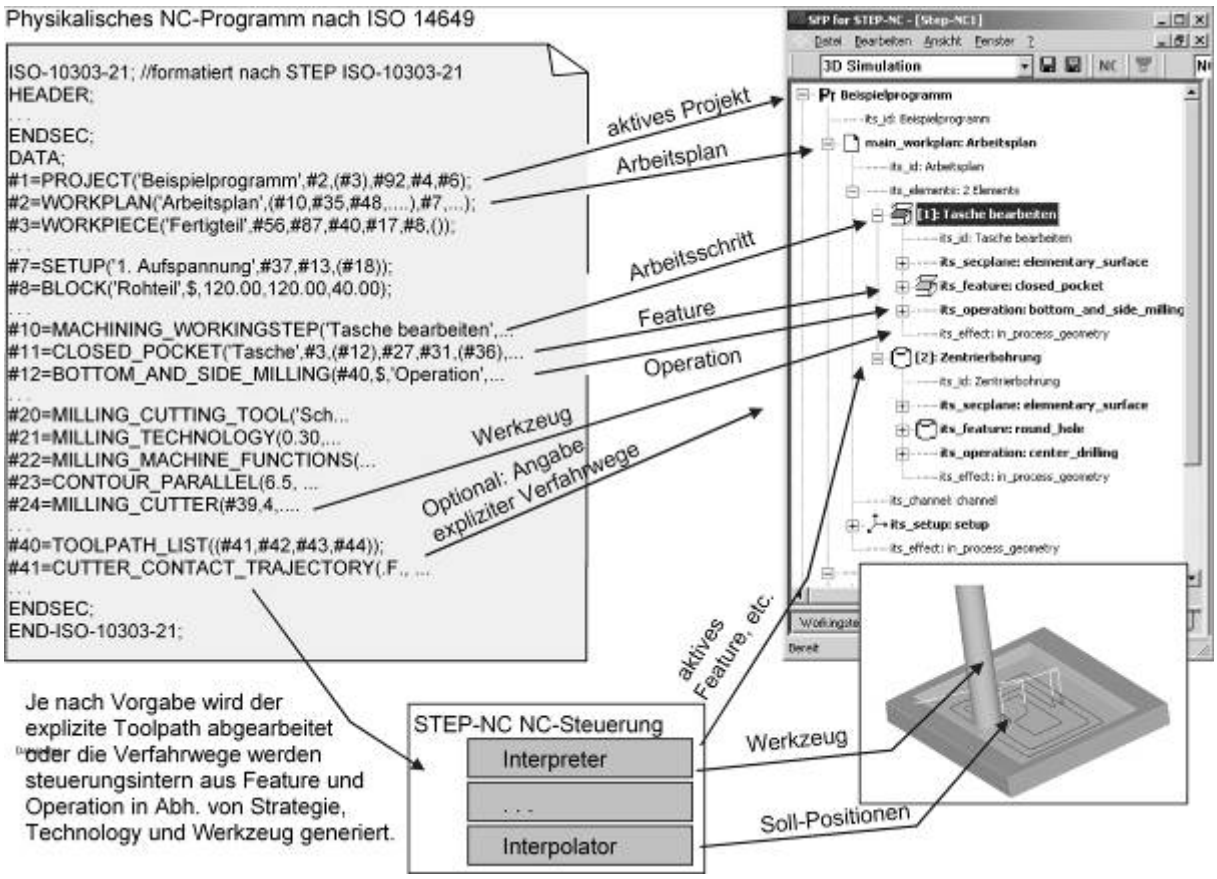


Bild 6.5: Informationsabstraktion vom STEP-NC *Workingstep* bis zum *Toolpath*
Level of information from *Workingsteps* to single *Toolpaths*

Die entwickelten und realisierten NC-Workingsteps dienen somit der

- durchgängigen, verlustfreien Abbildung der im NC-Programm kodierten Aufgaben
- Strukturierung der Programmumsetzung in anwendergerechten Abstraktionsebenen
- Auflösung in interaktive Teilaufgaben zur Abarbeitung in den folgenden NC-Modulen
- Bereitstellung von Prozessinformationen für die Prozessüberwachung und -regelung
- internen Weiterleitung von Fortschrittsmeldungen und Parametereingriffen
- durchgängigen Übernahme und dem Abgleich adaptierter Parameter
- Entwicklung neuer, anwenderspezifischer Abläufe zur Programmumsetzung

6.1.3. Softwaretechnisches Konzept

Software concept

Dem Entwickler wird mit den elementaren NC-Workingsteps ein Baukasten bereitgestellt, aus dem er eigene, anwendungsspezifisch optimierte Abläufe zusammensetzen kann. Dazu übernimmt er bestehende NC-Workingsteps, kombiniert diese zur Umsetzung seines spezifischen Prozesswissens und ergänzt sie durch eigene, spezialisierte NC-Workingsteps. Der Ansatz, unterschiedliche NC-Workingsteps zur Abbildung des NC-Programms und der internen Steuerungsfunktionen zu definieren, erfüllt auf diese Weise die Anforderung nach einer offen und strukturiert erweiterbaren NC-Programminterpretation und -verarbeitung. Der Entwickler wird durch die Wiederverwendung bestehender Lösungen bei der Optimierung und der Erweiterung des Funktionsumfangs aktiv unterstützt.

6.1.3.1. Strukturierte Kapselung der Programmelemente in Klassen

Encapsulation of the NC-program elements in software classes

Softwaretechnisch bietet es sich an, die einzelnen NC-Workingsteps auf eine objektorientierte Klassenstruktur abzubilden. Jedem NC-Workingstep entspricht dabei ein Objekt, dessen Attribute die Parameter des NC-Workingsteps implementieren und dessen Methoden zum einen den Zugriff auf die Attribute kontrollieren und zum anderen das Verhalten des Workingsteps beschreiben. Die sogenannte Vererbung oder Spezialisierung erlaubt es, die Attribute und Methoden gleichartiger NC-Workingsteps in einer Basisklasse zusammenzufassen. Indem der Entwickler seine individuellen NC-Workingsteps, bzw. deren Klassen von dieser Basisklasse ableitet, übernimmt er die darin bereits implementierten Attribute und Methoden. Sein Entwicklungsaufwand wird reduziert, Bild 6.6.

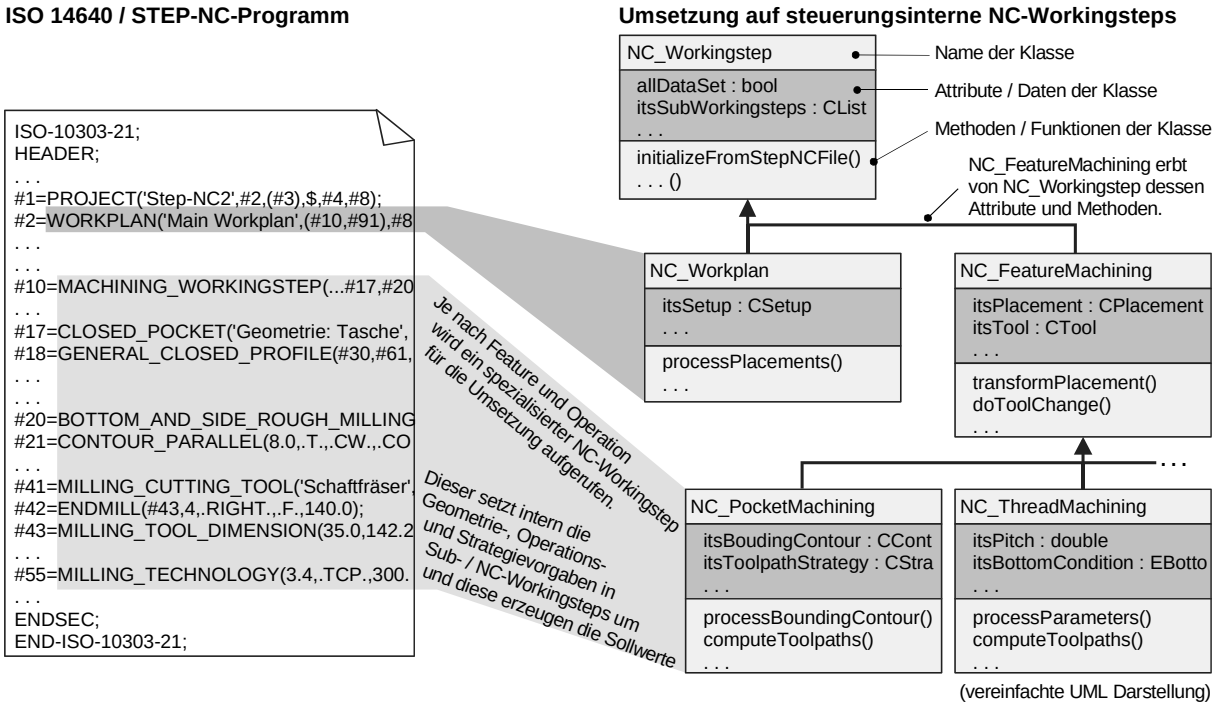


Bild 6.6: Spezialisierung der NC-Workingsteps durch Vererbung (in UML Notation)
Specializing NC-workingsteps by the means of heritage (UML noted)

Die grundlegenden NC-Workingsteps muss der Steuerungshersteller als Klassenbibliothek in seiner Steuerungssoftware implementieren und offen als Ausgangspunkt für Erweiterungen bereitstellen. Hierzu zählen insbesondere die von der NC-Steuerung direkt in Lage- und Sollwerte umzusetzenden, elementaren Schalt- und Wegbefehle der Steuerung. Sie bilden folglich die Schnittstelle zu den steuerungsspezifischen, vom Hersteller nicht weiter offengelegten Elementarfunktionen. Jede weitere, in die Steuerung zu integrierende NC-Workingstep-Klasse muss im Rahmen der Programm- und Geometrieverarbeitung auf diese Elementarobjekte heruntergebrochen werden.

Idealer Weise sind die Klassen der elementaren NC-Workingsteps bereits von einer gemeinsamen, ebenfalls durch den Steuerungshersteller vorgegebenen Basisklasse abgeleitet. Denn so kann der Hersteller festlegen, welche Methoden und Funktionen seine NC-Steuerung voraussetzt, damit die einzelnen Steuerungsmodule vom Interpreter über die Geometriekette bis hin zu den Sollwertschnittstellen mit den verschiedenen NC-Workingsteps interagieren können. Darüber hinaus muss die Basisklasse einheitliche Fehlermeldungen und Sicherheitsfunktionen implementieren.

Die Steuerungshersteller als auch externer Entwickler setzen auf dieser Struktur auf, um zusätzliche NC-Workingsteps zu ergänzen oder das Funktionsverhalten bestehender NC-Workingsteps entsprechend ihren Prozessmodellen zu spezialisieren. Hierzu werden entweder eigene, neue NC-Workingsteps von der Basisklasse „*NC_Workingstep*“ abgeleitet (z.B. „*NC_SpecialMachining*“), oder ein bestehender NC-Workingstep, wie z.B. die Auflösung einer Taschengeometrie, wird individuell überladen („*NC_MyPocket_Machining*“) und dem abzuarbeitenden STEP-NC-Programmelement „*Closed_Pocket*“ mithilfe einer Konfigurationstabelle anstelle der alten Implementierung („*NC_PocketMachining*“) zugewiesen.

Dieser objektorientierte Ansatz lässt sich sowohl auf die Abbildung einfacher (DIN 66025) als auch komplexer (ISO14649) NC-Programmierschnittstellen anwenden. Vor dem Hintergrund der praktischen Umsetzung ist das ein wesentlicher Vorteil, da die DIN-Programmierung heute Stand der Technik ist und ein Wechsel auf eine neue Programmierung nicht ohne Migrationsschritte erfolgreich am Markt umzusetzen ist. Zudem unterstützt er den Entwickler bei der Implementierung. Die aufeinander aufbauenden NC-Workingsteps leiten den Entwickler zu einer systematischen Softwareentwicklung an, deren Abläufe transparent zusammenhängen und deren Umfang sich strukturiert erweitern lässt. Aus der Summe der vom Steuerungshersteller und von externen Entwicklern implementierten NC-Workingsteps ergibt sich ein Baukasten aus dem neue und optimierte Abläufe, d.h. Sub-Workingsteplisten zusammengestellt werden können. Gemäß den Anforderungen lassen sich bestehende Entwicklungen wieder verwenden und auf der Basis der offen gelegten Schnittstellen erweitern. Das spezifische Ablaufwissen wird in den Objekt-Klassen gekapselt. Die darin enthaltenen Algorithmen werden über die Schnittstellenfunktionen der Objekte aufgerufen, so dass sie von Dritten genutzt werden können und gleichzeitig ihr softwaretechnischer Inhalt, d.h. das individuelle Prozesswissen, geschützt ist.

6.1.3.2. Extern gesteuerte Verarbeitung der NC-Workingsteps

Externally triggered processing of the NC-workingsteps

Die NC-Steuerung hat die Aufgabe, die einzelnen im NC-Programm enthaltenen Anweisungen in eine Sequenz ausführbarer Sollwerte umzusetzen und diese zeit-, weg- und funktionssynchron auszuführen. Wie in Kapitel 2.2 des Stands der Technik dargestellt, setzen die Steuerungshersteller hierzu unterschiedliche Konzepte ein.

Die *Sinumerik 840D* ist modular aufgebaut. Ihre Funktionsmodule verarbeiten die Programminformationen sequentiell und generieren als Ergebnis neue, steuerungsinterne Datenstrukturen. Diese leiten sie ablaufsynchron an das jeweils nachfolgende Modul weiter. Bezogen auf die entwickelten NC-Workingsteps gehen dabei die kontextbezogenen Informationen über Arbeitsschritt, Merkmal und Operation verloren. Folglich würden die Eingriffsoptionen mit fortschreitendem Programmvorlauf schrittweise von einem Zugriff auf die NC-Programmvorgaben hin zu elementaren Wegsegmenten und letztlich einzelnen Sollwerten reduziert.

Ebenfalls mit dem prozeduralen Ansatz der Bosch Rexroth *MTX* ist es nicht möglich, eine bereits vorverarbeitete NC-Workingstepliste umfassend zu adaptieren oder komplett zu ersetzen. Die in der *MTX* definierten Funktionslisten werden nacheinander abgearbeitet und greifen dabei ebenfalls nur auf die Ergebnisse der vorhergehenden Funktionen zu. Die Programmvorgaben und die internen Zwischenergebnisse sind datentechnisch nicht miteinander verknüpft, so dass sie ausgehend von einer Änderung keine Adaptionsroutinen starten können.

Das prozedurale Konzept bietet sich jedoch an, um den Abarbeitungsfortschritt der NC-Workingsteps zu steuern und im Fall von Eingriffen zurückzusetzen. Dazu werden die Prozeduren gezielt als Methoden in den NC-Workingstepobjekten implementiert. Die einzelnen Methoden lassen sich von außen aufrufen, so dass die Steuerungsmodule den Ablauf weiterhin aktiv steuern und mit den Ergebnissen arbeiten können. Anderenfalls müssten übergreifende Funktionen, wie die Werkzeugkorrektur, Transformation oder die Geschwindigkeits-

führung in umfangreichen NC-Workingstepstrukturen implementiert werden, anstatt sie analog zu heutigen Steuerungsarchitekturen als funktionsorientierte Module zu implementieren.

Nach dem Abschluss einer Methode wechselt das Objekt seinen Zustand, um die nächsten Methoden zum Aufruf freizugeben und die Einhaltung der Umsetzungsabfolge sicherzustellen. Der Vorteil dieses Konzeptes liegt darin, dass die Objekte neben den ursprünglichen Programmvorgaben auch die Zwischenergebnisse sowie den Abarbeitungsstand verwalten. Bei einer Störung oder einem externen Eingriff wird dieser Zustand ausgewertet und, wenn es der Abarbeitungsstand zulässt, zurückgesetzt. Die vorherigen Umsetzungsmethoden werden erneut aufgerufen und übernehmen die adaptierten Parameter.

In Bild 6.7 sind die Methodenaufrufe und deren Abfolge exemplarisch dargestellt. In einem ersten Schritt wird im NC-Programm die Aufgabe, eine Tasche auszufräsen erkannt. Daraufhin wird das NC-Workingstepobjekt *„NC_PocketMachining“* instanziiert ①, indem sein Konstruktor „*NC_PocketMachining(…)*“ aufgerufen wird. Dieser löst selbstständig oder getriggert durch die Steuerungsmodule das Laden ②, Prüfen ③ und Auflösen ④ der NC-Programmdaten aus. Um das Auflösen des Workingsteps gezielt zu steuern und die spezialisierten Algorithmen gemäß den Anforderungen zu kapseln, bieten sich private, in UML-Darstellung durch einen Schlüssel symbolisierte Objektmethoden ⑤, ⑥ an. Darüber hinaus stellt das Objekt gezielt Funktionen zum Prozesseingriff bereit ❶, die nach ihrem Aufruf die anderen Methoden sperren und je nach Eingriff die Abfolge erneut starten ②…⑥ können.

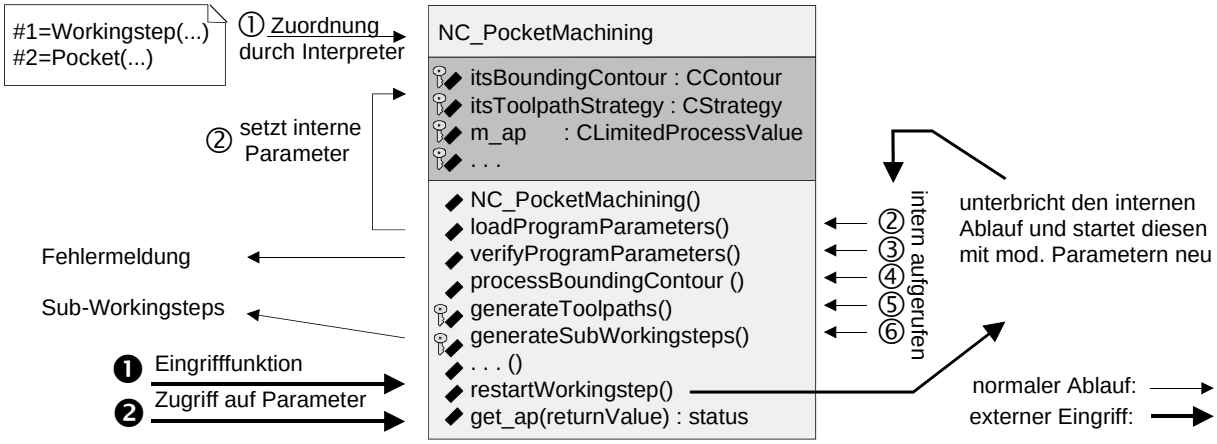


Bild 6.7: Beispiele für Grundfunktionen einer NC-Workingstep Klasse (vereinfachtes UML)
Exemplary base functions of a NC-workingstep class object (simplified UML)

Durch die Lösung die steuerungsinternen NC-Workingsteps in Objekte mit eigenständigen Methoden zu kapseln ergibt sich ein weiterer, für die Aufgabenstellung wesentlicher Vorteil. Die Objekte verwalten die Parameter der NC-Workingsteps in Form interner, geschützter Attribute und kontrollieren über Schreib- und Lesenfunktionen den Zugriff auf deren Inhalt, z.B. ❷ *„get_ap(returnValue) : status*“ in Bild 6.7. Somit können die NC-Workingsteps ihre Parameter untereinander abgleichen und gezielt auf Änderungen reagieren. Innerhalb der Objektmethoden wird geprüft, ob das Attribut von einem vorausgehenden Arbeitsschritt oder der laufenden Programmausrührung abhängt und inwieweit Änderungen zulässig sind. Untereinander tauschen die Workingsteps Interpretationsergebnisse, Randbedingungen oder Änderungen dynamisch aus oder sie stellen sie den Steuerungsmodulen zur Verfügung. Auf diese Weise prüfen die Objekte eigenständig, ob sich beispielsweise das Werkzeug zwischen zwei programmierten Arbeitsschritten ändert, die Technologieparameter umgestellt werden müssen, oder wie nach einem Prozesseingriff optimiert von einer Bearbeitungsstelle zu einer nachfolgenden, optimierten Bahn verfahren werden kann. Um nicht aufgrund der sich aus diesem Ansatz ergebenden, vielfältigen Möglichkeiten eine zu umfangreiche und komplexe

Anzahl an Methoden entwickeln zu müssen, werden übergreifende Funktionen in übergeordnete NC-Workingsteps verlagert oder innerhalb der Steuerungsmodule, wie dem Interpreter, implementiert.

6.2. Modular erweiterbare Programminterpretation

Modularly extendable NC-program interpretation

Der Interpreter hat die Aufgabe, den Inhalt des NC-Programms auf die beschriebenen NC-Workingsteps abzubilden und diese an die nachfolgenden Module des NC-Kerns zur weiteren Verarbeitung und Ausführung weiterzuleiten. Da konventionelle NC-Steuerungen während der Bearbeitung den Inhalt des NC-Programms für Zugriffe sperren und Prozesseingriffe nur über Steuerungsparameter, wie die Overrides für Vorschub oder Drehzahl zulassen, reagieren die Interpreter dieser Steuerungen nicht auf Programmänderungen oder Benutzereingriffe. Sie verarbeiten das NC-Programm rein sequentiell. Eine Bahnumplanung oder die Änderung von Technologieparametern sind nur nach einem Bearbeitungsabbruch und anschließender Programmkorrektur möglich. Diese konventionelle Programminterpretation und -verarbeitung wird durch das folgende Konzept ersetzt.

Mit dem Ziel, die Eingriffsmöglichkeiten in den laufenden Bearbeitungsprozess zu erweitern, wird ein Interpreter entwickelt, der mit der Sollwertausführung gekoppelt ist und über den die noch nicht umgesetzten Programmelemente bearbeitungsparallel adaptiert werden können. Hierzu bietet der Interpreter unterschiedliche Zugriffsebenen an, die auf den in Kapitel 6.1 vorgestellten NC-Workingsteps basieren. Aufgrund der durchgängigen Kopplung der NC-Workingsteps vom Interpreter über den NC-Kern bis zur Sollwertausführung wird sichergestellt, dass die Eingriffe synchron in die laufende Bearbeitung übernommen werden.

Zudem umfasst das Konzept die erforderlichen Basisfunktionen und Schnittstellen, mit denen die entwickelte NC-Programmierschnittstelle durchgängig steuerungsintern abgebildet wird. Dazu stellt der NC-Interpreter eine Bibliothek elementarer als auch aufgabenspezifischer NC-Workingsteps bereit. Diese lassen sich durch den Steuerungshersteller oder Prozessentwickler optimieren und um eigene NC-Workingsteps erweitern, (z.B. Modul „Flächen“ in Bild 6.8).

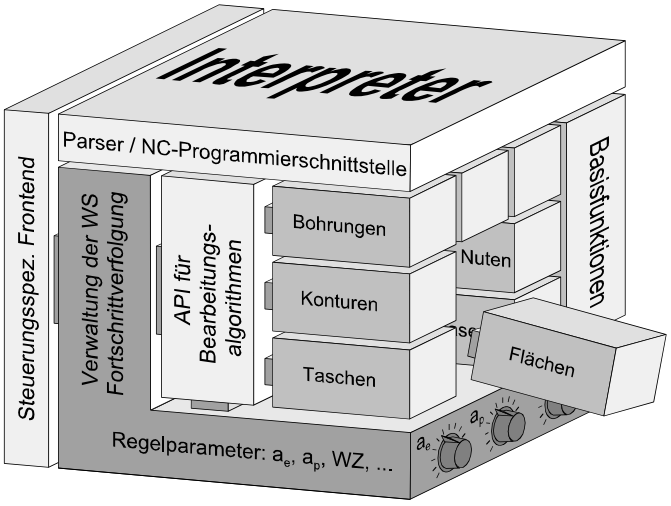


Bild 6.8: Interpreter mit Schnittstellen zur modularen Erweiterung um NC-Workingsteps
Interpreter with interfaces for modularly extending the NC-workingstep classes

Den Abarbeitungsfortschritt kontrolliert der Interpreter über die von ihm verwaltete NC-Workingstepliste. Des Weiteren steuert er fortschrittsabhängig den Umfang der zulässigen Programmeingriffe. Ein an die nachfolgenden Steuerungsmodule angepasstes Frontend, setzt

die NC-Workingsteps in Nachrichten um, die vom NC-Kern weiterverarbeitet werden. Die einzelnen für das Konzept im Interpreter erforderlichen Module und deren spezifischen Funktionen werden in den nachfolgenden Unterkapiteln erläutert.

6.2.1. Einlesen des NC-Programms und Zugriff auf dessen Daten

Parsing the NC-program and accessing its content

Das erste Teilmodul des Interpreters ist der sogenannte Parser zum Einlesen des NC-Programms. Seine Aufgabe besteht darin, die alphanumerisch kodierten Programmdaten auf der Basis einer vorgegebenen NC-Programmierschnittstelle zu analysieren und je nach Programmgröße schrittweise als Datenelemente in den Arbeitsspeicher der Steuerung zu laden. Im Speicher stehen die Programminformationen dann als zusammenhängende Datenstrukturen zur Verfügung und können vom Interpreter zur weiteren Verarbeitung ausgelesen werden. Mit dem Ziel, die Steuerung an unterschiedliche Befehlskodierungen und kundenspezifische Erweiterungen anpassen zu können, unterscheidet das Konzept zwischen der Datenhaltung des Parsers und der des Interpreters. Im Parser werden ausschließlich die Elemente des NC-Programms verwaltet, während der Interpreter den Programminhalt intern auf ein eigenes Datenmodell, die vorgestellten NC-Workingsteps, abbildet. Über Zeiger oder Identifikationsschlüssel sind die beiden Datenspeicher miteinander verknüpft. Schreib-, Lese- und Funktionsaufrufe stellen sicher, dass ein Datum nicht zeitgleich durch eine Teilfunktion manipuliert und eine andere ausgelesen werden kann. Der Interpreter kann jederzeit die von ihm benötigten Informationen nachladen. In der Gegenrichtung meldet der Parser Änderungen am NC-Programm an die NC-Workingstepverwaltung des Interpreters. Diese ruft nach jeder Änderung die betroffenen NC-Workingsteps auf, um deren Dateninhalt zu überprüfen. In Abhängigkeit von der erkannten Änderung werden einzelne Parameter eines NC-Workingsteps adaptiert oder daraus abgeleitete NC-Workingstepstrukturen verworfen und neu generiert.

Das Konzept unterscheidet drei Optionen, um in die NC-Programmvorgaben und deren Interpretation zur Laufzeit einzugreifen:

1. Über die Benutzerschnittstelle oder Regelsysteme kann kontrolliert auf die Datenhaltung des Parsers zugegriffen werden. So lassen sich die im NC-Programm kodierten Vorgaben manipulieren und vor ihrer Ausführung nachladen.
2. Der Interpreter bietet zu seinen NC-Workingsteps Schnittstellen an, über die sich vom Entwickler vorgegebene Parameter und Strategien ändern lassen.
3. Zusätzliche Parameter, wie Sicherheitsabstände, Abbruchstrategien oder Override in Abhängigkeit von den Eingriffsverhältnissen, gelten steuerungsweit und werden zentral als übergeordnete NC-Workingsteps oder Steuerungsvariablen in den offenen Schnittstellen der NC-Steuerung bereitgestellt.

Nachladen von Änderungen vor der Ausführung eines Arbeitsschritts

Im Parser wird die Anforderung, den Programminhalt bearbeitungsparallel zu manipulieren, realisiert, indem eingelesene Teile des NC-Programms erneut gelesen und mit den bereits im Speicher abgelegten Daten abgeglichen werden können. Hierzu arbeitet der Parser nicht einmalig und rein sequentiell. Zusätzlich speichert er die Quellposition der Vorgaben und liest diese gezielt vor der Abarbeitung nochmals ein. Dieser Ansatz hat jedoch den Nachteil, dass er Fehleingaben nicht ausschließen kann und es somit zur Laufzeit zu Störungen oder einem Bearbeitungsabbruch aufgrund von Datenfehlern kommen könnte. Daher wird diese Interaktionsmöglichkeit auf noch ausstehende, aktuell nicht in Abarbeitung befindliche Arbeitsschritte beschränkt. Bevor ein neuer Arbeitsschritt gestartet wird, gleichen die NC-Workingsteps über den Parser ihre Parameter mit dem aktuellen Datenstand des NC-Programms ab. Dabei wird durch den Parser die syntaktische Richtigkeit der gelesenen Programmdaten ge-

prüft. Ferner bewertet der Interpreter die Eingriffe ausgehend von den Prozessmodellen seiner NC-Workingsteps. In Abhängigkeit von den erkannten Fehlern hält der Interpreter die Programmausführung an, bietet dem Anwender die Möglichkeit zur Korrektur oder bricht die Bearbeitung sicher ab.

Ein Eingriff in die Programmvorgaben ist nicht zu beliebigen Zeitpunkten möglich. Mit dem Fortschritt der steuerungsinternen Programmverarbeitung entstehen verkettete NC-Workingsteps mit voneinander abhängenden Zusatzdaten. Sobald eine dieser NC-Workingsteplisten durch die NC-Steuerung weiter verarbeitet wird oder die darin enthaltenen Bewegungs- und Schaltanweisungen teilweise an der Maschine ausgeführt wurden, muss dieser Programmfortschritt bei den Eingriffen berücksichtigt werden. Alleine auf der Basis des NC-Programms sind jedoch die aus den Umsetzungsalgorithmen resultierenden Abhängigkeiten und Einschränkungen nicht nachzuvollziehen. Um Fehleingriffe zu verhindern, wird daher der Zugriff auf diejenigen NC-Programmdaten gesperrt, deren zusammenhängende Arbeitsschritte vollständig eingelesen sind und die sich in Abarbeitung befinden. In diesem Zustand ist ein Programmeingriff nur noch über die nachfolgend beschriebenen Softwarefunktionen auf der Basis der Datenhaltung des Parsers und den damit verknüpften, vom Interpreter generierten NC-Workingsteps sinnvoll.

Eingriffe in die Datenhaltung des Parsers und die NC-Workingsteps des Interpreters

Binden ein Regelsystem oder die Benutzungsoberfläche die Bibliothek des Parsers (vgl. Kapitel 5.1) mit ein, so ist ein gemeinsamer Datenzugriff über den Shared Memory des Parsers mit der NC-Steuerung möglich. Innerhalb der Zugriffsfunktionen werden die Eingaben softwaretechnisch geprüft und auftretende Fehler zurückgemeldet. Der Werker wird so direkt informiert und kann eine Fehleingabe korrigieren. In der laufenden Bearbeitung erfolgt die Übernahme der veränderten Programmvorgaben, indem der Parser an den Interpreter und die mit seinen Daten verknüpften NC-Workingsteps die Änderung meldet. Der NC-Workingstep prüft die Anwendbarkeit und adaptiert seine Attribute als auch die der ihm untergeordneten NC-Workingsteps. Bei komplexeren Eingriffen werden die betroffenen NC-Workingsteps neu umgesetzt.

Ebenso wie der Parser auf Änderungen reagieren kann, so ist es auch möglich, dass die vom Interpreter generierten und in dessen NC-Workingsteplisten verwalteten NC-Workingsteps ihre spezifischen Merkmale zur Prozessverfolgung und -manipulation nach außen zur Verfügung stellen. Diese Funktionen hängen jedoch von den implementierten Maschinen- und Bearbeitungsfunktionen ab. Dazu ist es sinnvoll, zwischen inneren und äußeren Eingriffsfunktionen zu unterscheiden:

Die **innere Eingriffsfunktion** eines NC-Workingsteps wird aufgerufen, wenn sich die Eigenschaften seiner Umgebung, d.h. die Attribute der mit ihm verknüpften, übergeordneten oder benachbarten NC-Workingsteps ändern. Durch diesen Ansatz lassen sich Technologieparameter ausgehend von einem Arbeitsschritt für alle ihm untergeordneten Sub-Workingsteps adaptieren. Zudem kann ein gerade ausgeführter NC-Workingstep sein Prozessergebnis oder die an ihm optimierte Einstellung an die ihm nachfolgenden, noch auszuführenden NC-Workingsteps weiterleiten.

Aufgrund der Vielzahl der verschiedenen NC-Workingsteps und ihrer Parameter können die Änderungen aber nicht direkt über Zugriffe auf die einzelne Prozessvariablen der NC-Workingsteps erfolgen. Anderenfalls würde die Weiterleitung stoppen, sobald ein NC-Workingstep diese Variable nicht kennt, da sie nicht zu seinem Prozessmodell gehört (z.B. eine Vorschubänderung an einem NC-Workingstep zum Palettenwechsel). Ein Lösungsansatz hierzu ist ein generisches Nachrichtensystem, in dem der Nachrichtentyp als auch die Adressaten über Identifier kodiert werden und die eigentliche Nachricht, d.h. die von dem Eingriff ausgehende Änderung, als neutraler Datenblock verpackt ist. Die Empfänger reagieren auf die Identifier, entpacken den Dateninhalt entsprechend dem erkannten Datentyp und setzen diesen

um. NC-Workingsteps, die von der Änderung nicht betroffen sind, leiten die Nachricht einfach der NC-Workingstepliste folgend weiter (vgl. Unterkapitel 6.3).

Die **äußeren Eingriffsfunktionen** erlauben, den Inhalt der NC-Workingsteps über eine externe Prozessregelung oder die Benutzungsoberfläche der NC-Steuerung zu verändern. Im Gegensatz zu den Override für Vorschub und Spindeldrehzahl sind sie jedoch nicht ständig verfügbar. Vielmehr werden sie in Abhängigkeit von dem jeweils aktiven oder ausgewählten NC-Workingstep als Eingriffsoption des Interpreters angeboten. Analog zu den am Rand heutiger Steuerungsoberflächen angeordneten Softbuttons sollte der Interpreter äußere Eingriffsfunktionen zu seinen NC-Workingsteps bereitstellen und diese in der Oberfläche kontextorientiert aktivieren. Das hierzu seitens der Benutzungsoberfläche notwendige Konzept wird in Kapitel 7 detailliert beschrieben.

Steuerungsschnittstellen für allgemeingültige Eingriffsparameter

Darüber hinaus sind viele Eingriffsparameter technologie- und maschinenspezifisch. Sie sollten daher zentral vorgehalten werden, so dass der Anwender als auch eine Prozessregelung hierauf einfach zugreifen können. Typische Parameter sind die heute über Override zentral manipulierbaren Vorschübe und Spindel-/Werkzeugdrehzahlen. Weitere, auf Basis des heutigen DIN-Codes nicht greifbare Parameter, leiten sich aus der entwickelten NC-Programmierschnittstelle und den darin enthaltenen Arbeitsschritten und Operationen ab. Als Beispiele seien hier das aktive Werkzeug, Sicherheitsebenen, operationsabhängige Override für Verbindungs-, Anrück-, Zustell- und Bearbeitungsbewegungen oder die Variation der Schnitttiefe und -breite bei der Fräsbearbeitung genannt.

Diese Schnittstellen stellen Eingriffsparameter als Steuerungsvariable zentral zur Verfügung. Die die NC-Workingsteps verarbeitenden Steuerungsmodule lesen diese Variablen aus und verrechnen sie mit den Vorgaben der NC-Workingsteps. Konzeptabhängig können die Eingriffe nach dem laufenden Arbeitsschritt zurückgesetzt werden oder aber programmweit gültig sein. Dem Anwender bietet dieser Ansatz ein einfaches Mittel, um gezielt technologische Vorgaben zu ändern, ohne das Programm eigens anhalten, korrigieren und neu starten zu müssen.

6.2.2. Abbilden des NC-Programms auf NC-Workingsteps

Mapping of the NC-program to the NC-workingsteps

Der Parser kann den Inhalt des NC-Programms nur einlesen, nicht jedoch logisch auswerten, um sie in eine Abfolge steuerungsinterner Datenobjekte umzusetzen, die dann schrittweise in Schalt- und Weganweisungen aufgelöst und als solche über die SPS und Antriebe ausgeführt werden. Diese Aufgabe übernimmt die Programminterpretation, indem sie die NC-Programmdaten auswertet und auf ihre internen Datenstrukturen abbildet. Im Kern dieser Abbildung stehen die in Kapitel 6.1 entwickelten NC-Workingsteps. Jeder dieser NC-Workingsteps repräsentiert eine typische Bearbeitungsaufgabe oder -funktion. Innerhalb des NC-Workingsteps werten dessen Methoden den eingelesenen Programminhalt aus und setzen ihn auf die eigenen, internen Attribute um.

Zuordnung der Programmschlüsselworte zu NC-Workingsteps

Die Zuordnung der vom Parser gelesenen Programmdaten zu einem NC-Workingstep kann beispielsweise über eine Zuordnungstabelle erfolgen, wie dies heute für die mnemonischen Codes der DIN Programmierung üblich ist. Dazu vergleicht der Interpreter Schlüsselworte, sogenannte Tags, mit einer Konfigurationstabelle und ruft bei Übereinstimmung eine hinterlegte Funktion im NC-Interpreter auf. Softwaretechnisch lässt sich ein solcher Auswahlmechanismus beispielsweise über Switches oder IF-Bedingungen implementieren. Dem Entwickler bietet dieser Ansatz die Freiheit, bestehende NC-Workingsteps durch eigene, speziali-

sierte Umsetzungsalgorithmen zu ersetzen. Ferner ist die Integration zusätzlicher NC-Workingsteps möglich, um Erweiterungen der NC-Programmierschnittstelle interpretieren und umsetzen zu können. Innerhalb des Interpreters werden die erforderlichen Grundfunktionen über eine offene Softwareschnittstelle, das sogenannte Application Programming Interface (API), bereitgestellt. Das API dient der Integration individueller NC-Workingstepobjekte mit den darin enthaltenen Bearbeitungsalgorithmen und deren Zuordnung zu den umzusetzenden NC-Programmelementen (vergl. „API für Bearbeitungsalgorithmen“ in Bild 6.8).

Je nach implementiertem Algorithmus kann ein NC-Workingstep eine einfache Verfahrbewegung bis hin zu einem komplexen Zyklus, bzw. einem STEP-NC-Feature-Workingstep interpretieren. Gegenüber der Zyklenauflösung heutiger Compiler erlaubt dieser Ansatz, dass die Interpretation zusammenhängend erfolgt und der Kontext zu den auszuführenden Bearbeitungsaufgaben bis in die Steuerung beibehalten wird. Datentechnisch wird das NC-Programm nur einmal zentral eingelesen. Es wird nicht von einem Präprozessor auf elementare Steuerungscodes heruntergebrochen, um dann von der NC-Steuerung sequentiell verarbeitet zu werden. Die NC-Steuerung erzeugt auch keine temporäre, für den Anwender nicht greifbare Zwischendatei. Im Datenspeicher des NC-Interpreters werden alle aus dem NC-Programm abgeleiteten NC-Workingsteps zentral verwaltet. Über ihre datentechnischen Verknüpfungen können sie untereinander Daten austauschen und Funktionen aufrufen. Schnelle Reaktionen auf Eingriffe als auch Datenmanipulationen und Neuberechnungen zur Bearbeitungsadaption sind ohne Programmunterbrechung und ein erneutes Kompilieren möglich.

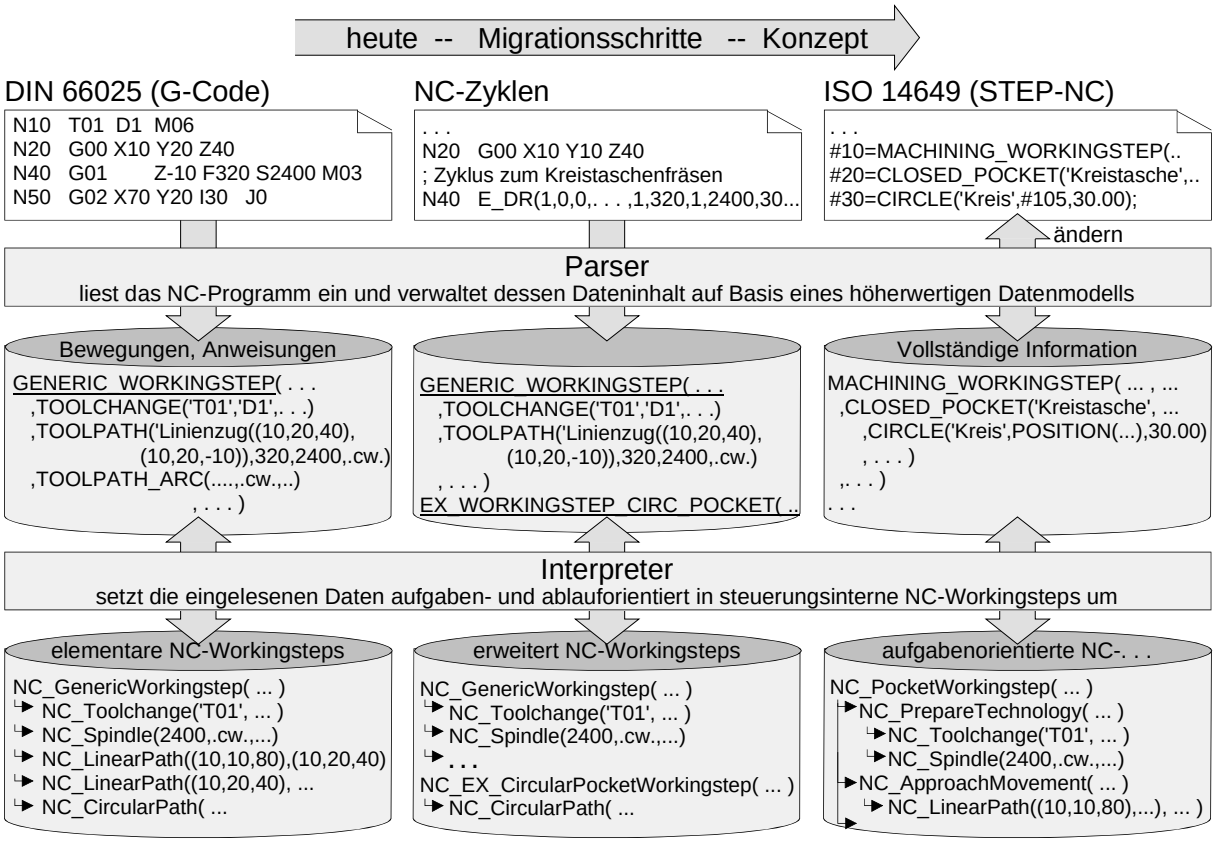


Bild 6.9: Erweiterung der Programmierung von G-Codes zu Workingsteps und Merkmalen
Extending the NC-programming from G-Codes to workingsteps and features

Zusätzlich zu der Programminterpretation steuern die NC-Workingsteps den weiteren Ladevorgang des Parsers. Dieses Konzept bietet den Vorteil, dass das NC-Programm aufgabenorientiert und nicht nur in Abhängigkeit von dem belegten Systemspeicher gelesen wird. Der Interpreter stellt sicher, dass der anstehende NC-Workingstep vollständig eingelesen und um-

gesetzt wurde, ehe dass die NC-Steuerung seine Abarbeitung und Ausführung beginnt. Dadurch kann der Interpreter die Bearbeitungsaufgabe in seinen spezialisierten NC-Workingsteps als Ganzes zusammenfassen, auswerten, ihren Abarbeitungsfortschritt verfolgen und nach außen geeignete Prozessinformationen sowie Eingriffsoptionen anbieten. Diese Lösung lässt sich direkt auf die in Kapitel 5.1 entwickelte NC-Programmierschnittstelle mit ihren Arbeitsplänen („*Workplan*“), NC-Funktionen („*NC-Function*“) und Arbeitsschritten („*Workingstep*“) übertragen.

Migrationskonzept von Befehlen über Zyklen hin zu Aufgabenbeschreibungen

Die geforderte Erweiterbarkeit der NC-Programmschnittstelle um neue Elemente erfolgt generisch auf der Basis der in Kapitel 5.1 beschriebenen EXPRESS-Beschreibungsdatei und der daraus zu kompilierenden Bibliothek. Eine syntaktische Änderung (z.B. DIN 66025 Code) bedingt dagegen zusätzlich eine neue Parserschnittstelle, die jedoch den Kern, d.h. die objektorientierte, in EXPRESS vormodellierte und mit dem Interpreter interagierende Datenstruktur weiter nutzt, indem sie nur die aus einem anderen Format einzulesenden NC-Programmdaten hierauf abbildet. Bezogen auf den NC-Code nach DIN 66025 wird dieser in EXPRESS als Subset aus einfachen Weg- und Schaltelementen modelliert und kann parallel mit dem umfangreichen ISO 14649 Datenmodell verwaltet werden. Dieser Ansatz erlaubt es, die Programmierschnittstelle und damit den Parser nicht nur um individuelle Steuerungscodes zu erweitern, sondern das Programmformat vollständig zu ersetzen. Eine Migration von der heute verbreiteten DIN-Programmierung über herstellerspezifische NC-Zyklen hin zu der im Rahmen dieser Arbeit mitentwickelten ISO 14649 ist möglich. Das entwickelte Datenmodell der ISO 14649 Programmierschnittstelle ist derart umfassend, dass neuere Projekte dieses Aufgreifen und G-Code Programme hierüber in die Planung zurückführen [SHIN05].

Die im Interpreter implementierten Algorithmen zur Ablauf- und Bahnauflösung können unabhängig von der physikalischen Programmierschnittstelle eingesetzt werden. Die Anpassung an die unterschiedlichen Programmformate erfolgt über den Parser, während der Interpreter über gleiche Zugriffsfunktionen auf eine einheitlich aufgebaute, je nach Programminhalt unterschiedlich detaillierte und um spezifische Datenelemente erweiterte Datenbasis zugreifen kann (Bild 6.9). Diese setzt er dann auf seine internen NC-Workingsteps um. Je nach verfügbarem Informationsgehalt erzeugt der Interpreter dabei elementare oder aufgabenorientierte NC-Workingsteps. Die hersteller- oder anwenderspezifischen Code-Erweiterungen lassen sich auf erweiterte NC-Workingsteps abbilden und weiterverarbeiten (im Bild 6.9 angedeutet mit „EX_“ für extended). Dies ist als Zwischenschritt zu betrachten, bis dass ein Standard einheitliche Definitionen anbietet und die Interpreterlogik diese bedienen kann.

Auch Datenbankkonzepte lassen sich von diesem Ansatz ausgehend umsetzen. Der Parser stellt die Datenbankanbindung bereit und übernimmt für den Interpreter die Zugriffe auf die Geometrie-, Ablauf- und Technologiedatensätze etc. Ein möglicher Anwendungsfall ist die Übernahme von Bearbeitungsmerkmalen aus der Planung, die Definition der Abarbeitungsfolge an der Steuerung, die Werkzeugauswahl aus dem Werkzeugmagazin der Maschine und die Zuordnung von Operationen und Technologiedaten aus einer Datenbank der Fertigung. Diese Flexibilität erlaubt es, das Konzept an beliebige NC-Programmformen, Datenbanken und Programmerweiterungen anzupassen. Die Anforderung, die Programmelemente um zukünftig standardisierte als auch auf spezifischem Wissen basierende Informationen und Abarbeitungsroutinen ergänzen zu können und in ein System zur durchgängigen Datenverwaltung zu integrieren, wird durch das Konzept gewährleistet.

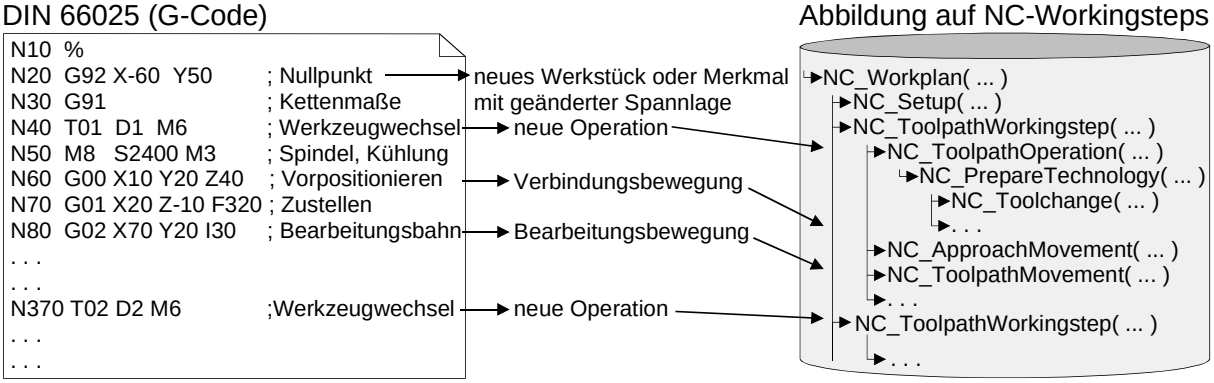


Bild 6.10: Interpretieren von G- und M-Codes und Abbildung auf einfache NC-Workingsteps
Interpretation of G- & M-codes and mapping to simple NC-Workingsteps

Ferner erlaubt die entwickelte Lösung die Integration intelligenter NC-Workingsteps, die auch in schwach strukturierten NC-Programmen Einzelbefehle analysieren und in Arbeitsschritten zusammenfassen. Somit lässt sich in einem DIN 66025 Programm aus einem NC-Zyklus oder einem Werkzeugwechsel der Beginn einer neuen, abgeschlossenen Aufgabe interpretieren und strukturieren. Beispielsweise werden im Fall einer einfach kodierten Bearbeitung nach dem Werkzeugwechsel die verschiedenen Weg- und Schaltanweisungen als NC-Sub-Workingsteps in einen NC-Machining-Workingstep geladen, bis ein erneuter Werkzeugwechsel oder eine erkennbare Änderung von Technologie und Strategie den Machining-Workingstep beenden. Über den Parser lädt der Interpreter dann die nächste abzubildende Programmanweisung und spiegelt auch diese an seiner NC-Workingstepauswahl. In der Folge entsteht gemäß der Aufgabenstellung aus einer schwach kontextorientierten Programmierung, wie der DIN-Programmierung, eine aufgaben- und technologieorientierte Datenstruktur (Bild 6.10). Auf dieser Grundlage kann eine werkergerechte Bearbeitungsverfolgung und -manipulation aufgebaut werden.

6.2.3. Zeitoptimierung durch parallele Vorverarbeitung

Reduction of processing time by parallel pre-processing

Das Reaktionsverhalten einer Steuerung hängt maßgeblich von der Geschwindigkeit ab, mit der Daten intern neu verarbeitet und adaptierte Sollwerte bereitgestellt werden können. Anstelle die einzelnen NC-Befehle sequentiell mit dem Abarbeitungsfortschritt zu verarbeiten, bieten sich auf der Grundlage des vorgestellten Konzepts der aufgaben- und ablauforientierten NC-Workingsteps eine parallele, vorausschauende Programmumsetzung an. Jeder der im NC-Programm enthaltenen NC-Workingsteps und den daraus zu generierenden NC-Workingsteps liest unabhängig die Vorgaben des NC-Programms ein und setzt diese soweit um, wie das ohne den Zugriff auf noch ausstehende, prozessabhängige Parameter der vorangehenden NC-Workingsteps möglich ist. So werden parallel zur laufenden Bearbeitung weitere NC-Workingsteps vorverarbeitet und stehen für eine schnelle Abarbeitung im Datenspeicher der Steuerung bereit. Zusätzlich lassen sich neben dem geplanten Prozessverlauf alternativ auszuführende Abläufe generieren. Diese können im Fall eines Eingriffs direkt aufgerufen werden, ohne die Zeit einer dann erst einsetzenden Berechnung alternativer Bahn- und Schaltsequenzen zu verlieren.

Dazu werden gesteuert durch den Abarbeitungsfortschritt und die beschriebenen Zustände Funktionen der NC-Workingsteps aufgerufen und in eigenen Threads ausgeführt. Beispielsweise lösen sich mit dem Programmstart die STEP-NC-Workingsteps „*Project*“, „*Workplan*“

und „*Workingstep*“ in deren steuerungsspezifische Datenobjekte *NC_Project*, *NC_Workplan* und *NC_Workingstep* auf. Diese Sub- Workingsteps beginnen wiederum unmittelbar mit der Umsetzung ihrer internen Parametervorgaben und ggf. der Erzeugung weiterer Sub-Workingsteps. Ausgehend von diesem Ansatz entsteht unmittelbar nach dem Öffnen des NC-Programms im Datenspeicher der NC-Steuerung eine verkettete, der programmierten Abarbeitungsfolge entsprechende Struktur von direkt ausführbaren NC-Workingsteps. Diese steuerungstechnisch vorverarbeiteten Anweisungen müssen dann nur noch dem Abarbeitungsfortschritt folgend aufgerufen und ausgeführt werden.

Hängt eine der Verarbeitungsroutinen von noch nicht bereitstehenden Ergebnissen vorangehender NC-Workingsteps, von Rückmeldungen aus dem realen Prozess oder von Werkereingaben ab, so bieten sich mehrere Optionen an, um auf die fehlenden Informationen zu warten. Zum einen kann der Thread sich selbstständig abbrechen und der NC-Workingstep wird in seinen vorhergehenden Zustand zurückgestuft. Sobald dann die ihn referenzierenden, vorangehenden NC-Workingsteps ihren Zustand ändern, wird der Thread erneut gestartet. Der Vorteil dieses Konzepts liegt in seiner einfachen softwaretechnischen Umsetzung. Jedoch wird der Thread auch gestartet, wenn die fehlenden Informationen ggf. noch nicht vorliegen. Daher ist ein gezielt ereignisgesteuerter Aufruf effektiver. Dazu registriert der wartende NC-Workingstep die an ihm aufzurufende Threadfunktion bei dem die Informationen bereitstellenden NC-Workingstepobjekt oder Softwaremodul. Ändern diese Objekte und Module ihren Zustand und folglich ihre Datenbasis, so werden je nach Softwarekonzept die wartenden Funktionen zur weiteren Verarbeitung freigegeben (Semaphoren) oder für den Fall, dass sie zuvor beendet wurden, als Rückruffunktionen (Callbackfunktionen) erneut aufgerufen. Damit werden die ausstehenden Operationen wartender NC-Workingsteps zeitnah zum Vorhandensein der Informationen fortgesetzt und abgeschlossen. Die NC-Workingsteps wechseln in ihre nachfolgenden Zustände. Damit geben sie die auf sie wartenden Threads frei, bevor sie selber mit ihrer nächsten Zustandsoperation fortfahren. Final lesen die SPS- und Antriebsschnittstellen der NC-Steuerung die generierten und gespeicherten Sollwerte aus den NC-Workingsteps aus, um sie auszuführen. Nach der finalen Abarbeitung werden die real ausgeführten Parameter zur Prozessanalyse ausgegeben und im Fall von Optimierungen oder Eingriffen an die nachfolgenden NC-Workingsteps übertragen. Der letzte Thread eines NC-Workingsteps wartet auf seine Freigabe bis er und seine Sub-Workingsteps vollständig von der Steuerung abgearbeitet wurden. Erst dann löscht er die ihm untergeordneten und seinen eigenen NC-Workingstep, so dass der von ihnen belegte Datenspeicher wieder freigegeben wird.

Dieses Konzept ist auf bestehende NC-Steuerungen übertragbar. In einer funktionsorientierten Steuerung, wie der Bosch MTX, würden die Datenobjekte der NC-Workingsteps im zentralen Speicher abgelegt werden. Die Jobliste entspräche dem gesteuerten Aufruf der NC-Workingstepmethoden. Der Zustandswechsel des NC-Workingsteps würde den Abschluss eines Bosch-Jobs widerspiegeln. Um jedoch im laufenden Prozess bereits weiterverarbeitete NC-Workingsteps adaptieren, d.h. neu berechnen zu können, müsste ausgehend von dem Eingriff in die Daten des NC-Workingsteps auch der erneute Aufruf der verarbeitenden Funktionen initiiert werden können. Dies wäre über zyklische Funktionen möglich, die fortlaufend Änderungen an den Daten verfolgen und Zugriff auf die Funktionsliste haben.

Einfacher ist die Umsetzung in einer bereits objektorientiert implementierten NC-Steuerung. Softwareobjekte bieten die Möglichkeit ihre Attribute, d.h. die objektspezifischen Daten, hinter Zugriffsfunktionen zu kapseln. Somit ist eine Änderung nur über einen definierten Funktionsaufruf möglich. Ein Beispiel für eine objektorientiert programmierte NC-Steuerung ist die Sinumerik 840D. Die Einsprungfunktionen der Sinumerik 840D lassen sich als Zustände interpretieren. Zwischen diesen Zuständen kann in Compilezyklen das NC-Programm in Form von NC-Workingsteps verarbeitet und in einer zentralen Datenbasis nachgehalten werden.

6.2.4. Fortschrittsteuerung auf Basis der NC-Workingsteps

Step wise program execution based on NC-Workingsteps

An heutigen NC-Steuerungen kann der Werker verschiedene Betriebsarten, wie Automatik, Einzelsatz im Automatikbetrieb (SBL), Einzelsatzeingabe (MDA), Referenzpunktfahrt (REF) oder Handbetrieb (JOG) anwählen. In Abhängigkeit von der Betriebsart wechselt die Steuerung in einen spezifischen Zustand, in dem die einzelnen Bedienfunktionen gesperrt oder freigegeben werden. Darüber lassen sich Zugriffskonflikte ausschließen, wie sie durch eine gleichzeitige manuelle Bedienung und automatische Programmausführung entstehen würden.

Die Forderung in der Betriebsart Einzelsatz (SBL) technologisch zusammenhängende Arbeitsschritte nach gezielten Kriterien schrittweise auszuführen, ist auf der Grundlage der vorgestellten NC-Workingsteps und ihrer vom Interpreter gesteuerten Abarbeitung möglich. Neben der konventionellen Unterbrechung nach einzelnen NC-Befehlen kann der Facharbeiter die Abarbeitung definiert beim Erreichen technologischer Kriterien anhalten. Die verschiedenen Kriterien definiert der Entwickler im Rahmen seiner NC-Workingsteps. Sie werden in den steuerungsinternen Nachrichtenfluss als Einsprungmarken eingefügt und dem Facharbeiter zur Auswahl in der Benutzeroberfläche angezeigt. Innerhalb der NC-Steuerung vergleichen die Module und Prozeduren die zu verarbeitenden Nachrichten mit den angewählten Kriterien und warten bei Übereinstimmung bis zur Quittierung durch den Facharbeiter. Dieses Konzept erlaubt beispielsweise die Bearbeitung gezielt zwischen zwei Arbeitsschritten, am Ende einer Bearbeitungsbewegung oder nach dem Erreichen einer Rückzugsebene zu unterbrechen. Ein gleichmäßiger Schnittprozess wird dadurch zusammenhängend ausgeführt und nicht, wie im G-Code üblich, fortlaufend aufgrund der Grenzen seiner Wegsegmente und Schaltbefehle unterbrochen. Bei Bedarf sind jedoch auch diese elementaren Unterbrechungen auf Befehls- und Segmentniveau möglich. Hierzu werden dann anstelle von NC-Workingsteps, Zustellebenen oder Werkzeugwechseln elementarere NC-Workingsteps, wie Linien-, Kreisbogen- und Kurvenzugabschnitte oder Schaltfunktionen als Einsprungmarken definiert.

6.3. Steuerungsinterne Kommunikation der NC-Workingsteps

Communication among the NC-Workingsteps

Die Interpretation der NC-Programmvorgaben und ihre Umsetzung auf NC-Workingsteps lassen sich nicht direkt zur Ansteuerung der Maschinenachsen und -funktionen nutzen. Zuvor müssen die NC-Workingsteps steuerungsintern weiterverarbeitet und in synchronisierte Sollpositionen, Schaltanweisungen und zu überwachende Prozesszustände aufgelöst werden. Dazu werden sie an spezialisierte Steuerungsfunktionen übergeben, d.h. kommuniziert. Das hier entwickelte Konzept erfordert, dass die Änderungen am NC-Programminhalt als auch an den daraus abgeleiteten NC-Workingsteps durchgängig möglich sind. Hierzu ist es notwendig, dass nicht nur der Inhalt sondern auch der Zusammenhang zwischen den NC-Workingsteps bis zu den final auszuführenden Sollwerten ausgetauscht und nach Eingriffen adaptiert werden.

6.3.1. Kommunikationskonzepte

Concepts for communication

Zum Austausch der vom Interpreter in Listen gespeicherten und untereinander verknüpften NC-Workingsteps bieten sich in Abhängigkeit von dem Aufbau der dem Interpreter nachfolgenden Steuerungsmodule unterschiedliche Kommunikationskonzepte an (Bild 6.11).

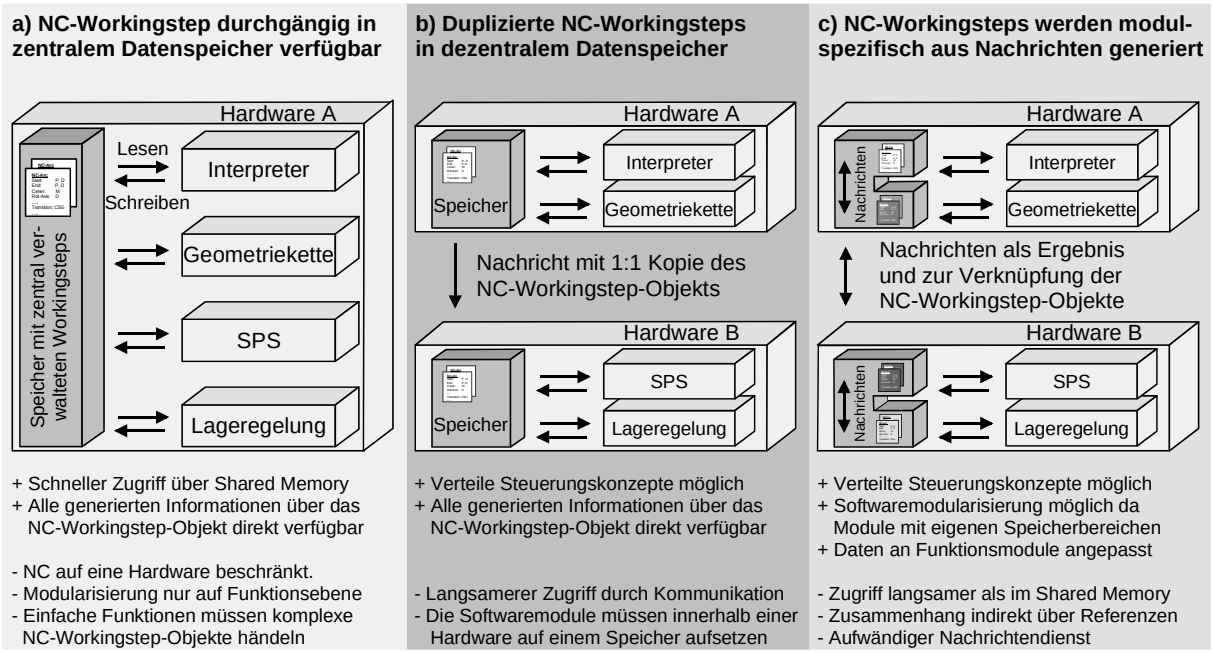


Bild 6.11: Konzepte zur Verwaltung der NC-Workingsteps im Steuerungsspeicher
Concepts to manage the NC-Workingsteps in the numerical control's memory

NC-Workingsteps als durchgängige Datenobjekte in einem zentralem Speicher

Die datentechnisch schnellste Kommunikationslösung basiert auf einem gemeinsam von allen Steuerungsmodulen genutzten Speicherbereich. In diesem werden die NC-Workingstepobjekte zentral abgelegt und können sequentiell von den einzelnen Steuerungsmodulen und den darin implementierten Funktionen aufgerufen und weiterverarbeitet werden (Bild 6.11 a). Die Kommunikation zwischen den Funktionen und Modulen erfolgt über Zeiger, die die Speicheradresse des auszutauschenden NC-Workingstepobjekts enthalten. Die von den Steuerungsfunktionen generierten Ergebnisse werden als Attribute in den bestehenden Datenstrukturen der NC-Workingsteps gespeichert oder als Sub-Workingsteps in die ablaufsynchrone NC-Workingstepliste eingefügt. Durch das Setzen von Attributen wird der Abschluss eines Funktionsschritts gemeldet und der NC-Workingstep wird für die nachfolgende Verarbeitungsfunktion freigegeben. Die Anforderung, dass ein NC-Workingstep auf den Ergebnissen der vorangehenden Bearbeitung aufbaut und darüber synchronisiert wird, ist somit realisiert. Dieser Ansatz ließe sich speichertechnisch auf heutige NC-Steuerungen, wie der Bosch-Rexroth MTX, der Heidenhain TNC Familie als auch auf die Siemens *Sinumerik 840D* übertragen, da ihre NC-Kerne auf einer Hardware einen gemeinsamen Speicher nutzen.

Neben der schnellen Kommunikation liegt ein weiterer Vorteil eines zentralen Speichers in der Koordination der nach einem externen Eingriff neu oder wiederholt auszuführenden Steuerungsalgorithmen. Die betroffenen NC-Workingsteps analysieren dazu ihre internen Daten und bestimmen, welche bereits ausgeführten Steuerungsfunktionen aufzurufen oder zu wiederholen sind. Dazu ändern sie ihren Abarbeitungszustand rückwärtig und löschen die zugehörigen Umsetzungsergebnisse. Wird der Umsetzungsablauf durch die NC-Workingsteps und ihre Zustände gesteuert, so werden in Abhängigkeit vom Zustand des NC-Workingsteps automatisch die jeweiligen Funktionen des Interpreters, der Werkzeugkorrektur, der Transformation, der Geschwindigkeitsführung etc. ablaufsynchron aufgerufen und ausgeführt.

Duplizierte NC-Workingstepobjekte in modulspezifischen Datenspeichern

In einer verteilten Steuerung, deren Teilfunktionen auf mehreren Hardwareplattformen implementiert sind (Bild 6.11 b) oder deren Softwaremodule separate Speicherbereiche nutzen

(Bild 6.11 c), ist kein durchgängiger gemeinsamer Zugriff auf die NC-Workingstepobjekte möglich. Stattdessen instanziiert jedes Modul die NC-Workingsteps in seinem individuellen Speicher neu. Zur Kommunikation der NC-Workingsteps müssen diese als Datenpakete verpackt und physikalisch zwischen den Plattformen und Modulen ausgetauscht werden. Diese Art der Datenkommunikation ist gegenüber dem zuvor beschriebenen, direkten Datenzugriff zeitintensiver.

Eine mögliche Umsetzung, um die Daten eines NC-Workingsteps über mehrere Hardwareplattformen auszutauschen, besteht darin, die Datenstrukturen der NC-Workingsteps in Nachrichten zu kopieren und auf der Empfängerplattform ein Duplikat der NC-Workingsteplisten anzulegen (Bild 6.11 b). In Abhängigkeit von dem Umfang der NC-Workingsteps und ihren Parametern müssten dazu große Datenmengen ausgetauscht werden. Vor dem Hintergrund einer schnellen Reaktion auf Eingriffe bietet sich dieser Ansatz nur für einfache NC-Workingstepstrukturen und Steuerungen mit wenigen Modul- oder Speicherübergängen an.

Aufgabenoptimierte NC-Workingstepobjekte in modulspezifischen Datenspeichern

Um bei komplexeren Strukturen den auszutauschenden Datenumfang gering zu halten, bietet es sich an, die NC-Workingsteps und ihren Inhalt entsprechend den Anforderungen der nachfolgenden Steuerungsmodule zu reduzieren. Hierzu werden nur die zur weiteren Umsetzung und Ausführung notwendigen Informationen kommuniziert (Bild 6.11 c). Beispielsweise sind für die Transformation der kartesischen Bearbeitungsbahnen auf die maschinenspezifischen Achsbewegungen die im NC-Programm kodierten Bearbeitungsschritte, Merkmale und Operationen nicht mehr unmittelbar relevant. Stattdessen reicht es aus, die vom Interpreter erzeugten und von der Werkzeugkorrektur adaptierten Raumkurven des Werkzeugbezugspunkts sowie die bahnsynchron anzuwendenden Technologieparameter auszutauschen.

Daher wird das NC-Workingstepkonzept um weitere, für die jeweiligen Steuerungsmodule spezifisch reduzierte NC-Workingsteps ergänzt. Jedes Modul generiert als Ergebnis des in ihm lokal gespeicherten und verarbeiteten NC-Workingsteps eine Kommunikationsnachricht. Diese Nachricht enthält alle relevanten Daten, damit im nachfolgenden Modul ein neuer, an seine datentechnischen und funktionalen Anforderungen angepasster NC-Workingstep initialisiert werden kann. Praktisch kann eine solche Nachricht als Datenblock über heute bereits in NC-Steuerungen verwendete Kommunikationsmechanismen, wie NC-DDE, OPC, OSACA oder CORBA ausgetauscht werden.

Zusätzlich enthält jede zur Weiterverarbeitung eines NC-Workingsteps versendete Nachricht Daten zur Synchronisation, Forschrittsverfolgung und, wie später in Kapitel 6.5 ausgeführt, Vorgaben zur Prozessüberwachung und vorausschauenden Prozessadaption. Ein softwaretechnischer Lösungsansatz sind Identifier, d.h. eindeutige Zahlenschlüssel, mit denen der Zusammenhang zwischen den ursprünglichen, übergeordneten und den modulspezifisch vereinfachten NC-Workingsteps hergestellt wird. Ausgehend von diesen Identifiern kommunizieren die NC-Workingstepobjekte später untereinander, um ihren Abarbeitungszustand, Änderungen oder Unterbrechungen bidirektional auszutauschen. Der Kommunikationsaufwand wird dadurch auf den Identifier sowie die Nachricht über die geänderten Daten reduziert.

Puffer zur synchronen Kommunikation der NC-Workingsteps

Werden die NC-Workingsteps zentral für alle Steuerungsmodule in einem gemeinsam genutzten Speicherbereich verwaltet, lässt sich die Abfolge der an ihnen auszuführenden Verarbeitungsroutinen über ihre Zustände und die Logik der einzelnen Module koordinieren. Nach dem Abschluss einer Routine wird der Zugriff, d.h. der Zeiger auf ein NC-Workingstepobjekt an das jeweilige Steuerungsmodul zurückgegeben. Dieses prüft anhand des Zustands des NC-Workingsteps, ob die letzte Funktion erfolgreich war und übergibt das Objekt zur weiteren Verarbeitung an die nächste Routine seiner Funktionsliste oder die des nachfolgenden Steuerungsmoduls.

Sind die Module durch separate Datenspeicher voneinander getrennt, bzw. tauschen sie Nachrichten untereinander aus, so muss die Kommunikationsplattform die Reihenfolge ihrer Abarbeitung koordinieren. Dazu werden die Nachrichten in Abhängigkeit von der geplanten Reihenfolge versendet und vom Empfängermodul in derselben Abfolge weiterverarbeitet. Des Weiteren hat die Kommunikationsplattform die Aufgabe, Verzögerungen, die aus den unterschiedlichen Verarbeitungsgeschwindigkeiten und Funktionsumfängen innerhalb der einzelnen Module resultieren, auszugleichen.

Diese Verzögerungen und die speichertechnische Trennung schränken zudem die Reaktionsfähigkeit bei der Umsetzung von Prozesseingriffen ein. Nach einem Eingriff müssen alte Vorgaben und NC-Workingsteps über mehrere Module hinweg gelöscht oder manipuliert werden. In einer rein sequentiellen Nachrichtenkette müssten dazu erst alle vorangehenden Nachrichten abgearbeitet oder gelöscht werden, ehe dass auf eine Änderung reagiert werden könnte. Günstiger wäre es, wenn die Eingriffsinformationen als eigenständige Nachrichten die ursprüngliche Nachrichtenabfolge überholen und von den Modulen zur Manipulation der alten Nachrichten ausgewertet eingesetzt werden könnten.

Ein einfacher Mechanismus, der diesen Anforderungen gerecht wird, existiert durch unterschiedlich gewichtete Datenpuffer. In diesen werden die Nachrichten entsprechend der Reihenfolge ihres Eintreffens zwischengespeichert und in Abhängigkeit von ihrer Ausführungspriorität ausgelesen (Bild 6.12). So wird zum einen die Abarbeitungsreihenfolge beibehalten (FiFo, First in First out) und zusätzlich ist ein Überholen von hoch prioren Nachrichten möglich. Eine solche priore Nachricht ist beispielsweise eine arbeitsschrittabhängige Programmunterbrechung oder der Eingriff in bereits gepufferte NC-Workingsteps.

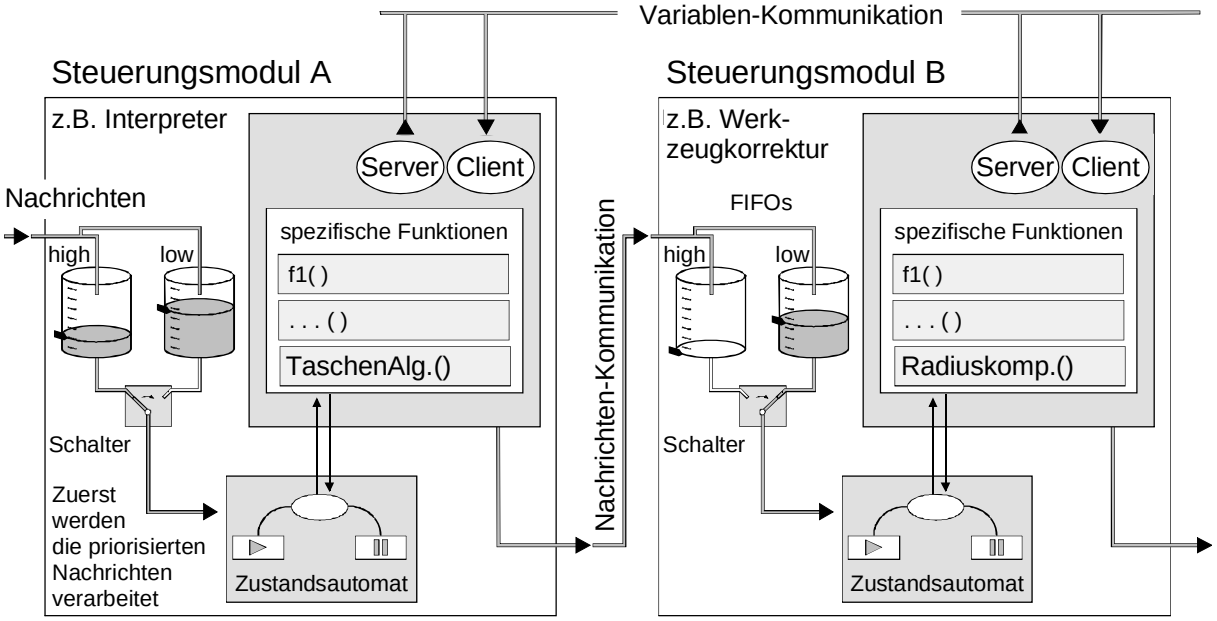


Bild 6.12: Nachrichtenaustausch zwischen verschiedenen Softwaremodulen über Pipelines
Pipe based message communication between the different software modules

6.3.2. Nachführen des Bearbeitungsfortschritts

Tracing the progress of the machining process

Sobald ein NC-Workingstep als Nachricht an ein nachfolgendes Modul kommuniziert und dort als ein neuer NC-Workingstep instanziiert wurde, könnte er gelöscht werden. Dabei gingen jedoch die modulspezifischen, nicht weitergeleiteten Informationen und der Zusammen-

hang zu den übergeordneten NC-Workingsteps verloren. Um im Fall eines Prozesseingriffs schnell reagieren zu können, ist es daher sinnvoll, die einzelnen NC-Workingsteps und die aus ihnen hervorgegangenen Listen solange zu speichern, bis dass sie für Parameterübernahmen und Eingriffe in laufende oder in nachfolgende Arbeitsschritte nicht mehr relevant sind. Dazu melden die einzelnen Steuerungsmodule vordefinierte Abarbeitungszustände an die vorgelagerten Module und insbesondere an die Fortschrittsüberwachung, die je nach Steuerungsmodus den Abarbeitungsfortschritt der NC-Workingsteps unterbricht oder freigibt, zurück.

Die gemeldeten Zustände werden in den übergeordneten Modulen ausgewertet. Insbesondere der Interpreter ist auf die Rückmeldung angewiesen, um Ist-Positionen und Ist-Werte in die Auflösung der nachfolgenden NC-Workingsteps zu übernehmen. Die Zustände werden somit zur Steuerung der an den NC-Workingsteps auszuführenden Funktionen eingesetzt. Im letzten Schritt melden sie, dass alle Abhängigkeiten aufgelöst und abgearbeitet wurden, so dass der NC-Workingstep gelöscht werden kann. Zudem werden sie genutzt, um das interne Prozessabbild der NC-Workingsteplisten zu aktualisieren. Über diesen Ansatz lässt sich erreichen, dass ein Eingriff direkt an die laufenden oder die noch auszuführenden NC-Workingsteps weitergeleitet wird. Folglich kann in der Steuerung gezielter und somit schneller reagiert werden.

Aufgrund der Vielzahl unterschiedlicher NC-Workingsteps und deren entwicklerspezifischen Erweiterungen, bietet es sich an, dass die NC-Workingsteps die Auswertung der Fortschritts- und Eingriffsnachrichten individuell übernehmen. Ausgehend von dem Ansatz einer verketteten NC-Workingstepliste wird jede Nachricht schrittweise weitergereicht, bis ihr Identifier von einem der NC-Workingsteps erkannt wird. Der so adressierte NC-Workingstep reagiert, indem er seinen Zustand anpasst und damit die für ihn geltenden Eingriffsmöglichkeiten ändert. Mögliche Zustände der NC-Programmverarbeitung sind „*preparing*“ während der Programminterpretation, „*prepared*“ nach der Auflösung in Sub-Workingsteps und der Berechnung von Verfahrwegen, „*processed*“ nach der Übergabe an die Geometrieverarbeitung, „*adapted*“ nach der bearbeitungsgesteuerten Anpassung oder „*executed*“ nachdem der NC-Workingstep von der Maschine ausgeführt wurde. Jeder Zustand beschreibt eine definierte Situation des NC-Workingsteps. Erst nach dem Erreichen eines Zustandes, wird der NC-Workingstep für die nachfolgende Operation freigeben. Ferner stellen die Zustände sicher, dass auf Zwischenergebnisse oder nicht abgeschlossene Manipulationen nicht unkontrolliert zugegriffen werden kann. Anderenfalls könnte die Endposition einer im Interpreter neu generierten Bewegungsbahn vor ihrer finalen Berechnung ausgelesen werden und die darauf aufbauenden Anschlussbewegungen fehlerhaft sein.

6.3.3. Schnittstellen zur Kommunikation von Eingriffen

Interface for communicating manipulations

Das Konzept, auf der Ebene der NC-Workingsteps einzugreifen, erfordert definierte Schnittstellen nach außen. Dabei sind aufgrund der offenen Programmierschnittstelle sowie den erweiterbaren NC-Workingsteps weder der Umfang der möglichen Eingriffe noch die daraufhin auszuführenden Reaktionen starr vorgegeben. Ausgehend von der entworfenen modularen Erweiterbarkeit definiert der Entwickler, welche Eingriffe er in Abhängigkeit von dem aktuellen Bearbeitungsfortschritt zulässt und nach welcher Strategie die bereits generierten, untergeordneten NC-Workingsteplisten, ihr Dateninhalt und final die daraus resultierenden Sollwerte adaptiert, neu berechnet oder vollständig ersetzt werden. Teilweise muss dazu nur der Interpreter reagieren und seine NC-Workingsteps adaptieren. Haben die Eingriffe jedoch Einfluss auf die zeitnahe oder aktuell ablaufende Sollwertausführung, so müssen die Änderungen über die beschriebenen Kommunikationsstrecken hinweg geprüft und umgesetzt werden.

Softwaretechnisch lassen sich die manipulierbaren Programm- und NC-Workingstepparameter über steuerungsspezifische Schnittstellen an zentraler Stelle als Variablen offen legen. Beispiele sind die in der Siemens 840D verwendeten proprietären Bindings oder offene Standards, wie Microsoft DDE, OPC und nicht zuletzt die OSACA Schnittstellenmechanismen. Innerhalb des Interpreters werden die einzelnen Variablen entweder direkt von den NC-Workingsteps bedient und abgerufen oder auf Nachrichten zur Kommunikation mit der verketteten NC-Workingstepstruktur abgebildet. Betrachtet man die Benutzerschnittstelle und die Prozessüberwachung als Clients, so sind die NC-Workingsteps die Serverobjekte, von denen die zu manipulierenden Parameter bereitgestellt werden. Da die NC-Workingsteps erst in Abhängigkeit von den NC-Programmvorgaben sowie der Programmumsetzung zur Laufzeit instanziiert werden, muss der Interpreter die Schnittstellenparameter ebenso dynamisch anbieten oder aber zentral vorhalten. Die externen Benutzer- und Regelmodule verbinden sich in Abhängigkeit von der Bearbeitungsaufgabe, die ebenfalls vom Interpreter gemeldet wird, mit den Schnittstellen und bieten somit kontextorientierte Eingriffsmethoden an.

Typische Variablen sind der Wechsel in den Einzelsatzmodus, das einzusetzende Werkzeug, ein operationsabhängiger Vorschub, die Höhe einer Sicherheitsebene oder die Schnitttiefe. Sie leiten sich direkt aus den Elementen der NC-Programmierschnittstelle als auch den Parametern der implementierten NC-Workingsteps ab. Es bieten sich daher die gleichen generischen Ansätze zur Implementierung der Variablenstrukturen an, wie sie bereits mit der Programmierschnittstellenbibliothek konzeptioniert wurden.

Die Eingriffsschnittstellen werden von der NC-Steuerung abgefragt und bei der Verarbeitung und Ausführung der NC-Workingsteps umgesetzt. Um zu verhindern, dass die Systemressourcen durch ein fortlaufendes Abrufen der Schnittstellen, das sogenannte Polling, belastet werden, sollten die externen Schnittstellen Variablendienste anbieten. Hierzu bieten sich zwei Mechanismen an:

- Nachrichtenobjekte
- Eventmechanismen

Nachrichtenobjekte

Die Änderung einer externen Schnittstellenvariablen wird in einer Nachricht verpackt und als Datenobjekt an den aktuell aktiven, übergeordneten NC-Workingstep übergeben. Dieser entscheidet, ob er die Änderung umsetzt, sie an seine über- und untergeordneten NC-Workingsteps weitergibt oder ggf. zusätzliche neue Nachrichten generiert. Aufgrund der Verkettung gelangen die Nachrichten schnell an alle nachfolgend auszuführenden NC-Workingsteps und die sie bereits verarbeitenden Module. Dieser Mechanismus ist insbesondere dann vorteilhaft, wenn komplexere Datenstrukturen vor dem Austausch interpretiert, weiterverarbeitet und gesteuert ausgeführt werden müssen. Ein möglicher Anwendungsfall ist ein extern veranlasster Werkzeugwechsel: Beginnend mit der Nachricht mit der Beschreibung des einzuwechselnden Werkzeugs, bestimmt der übergeordnete, den Arbeitsschritt steuernde NC-Workingstep, wann ein Wechsel möglich ist und generiert bedarfsorientiert zusätzliche Freifahrbewegungen, Nachrichten zum Spindel- und Kühlschmierstoffstop sowie Anweisungen, um das Werkzeug zu wechseln und an die Bearbeitungsstelle zuzustellen, um dann mit den in Abhängigkeit von den neuen Werkzeuggeometriedaten korrigierten Bearbeitungsbahnen fortzufahren.

Eventmechanismen

Andere Eingriffe sind weniger komplex. Sie bilden typische Technologieparameter ab. Mit dem Ziel die Abarbeitung schnell zu adaptieren, müssen sie von den einzelnen Steuerungsmodulen und den darin gepufferten NC-Workingsteps möglichst direkt übernommen werden. Ein Nachrichtenprozess, der zuerst im Interpreter ausgelöst, dann durch dessen NC-Working-

stepliste und die der nachfolgenden Module laufen muss, bevor er auch an den Sollwertschnittstelen ausgewertet werden kann, wäre hierfür zu umfangreich und damit zu langsam.

Ein Lösungsansatz sind Events, ein Mechanismus, der auf eine sich ändernde Schnittstelle reagiert und direkt Funktionen auslöst. Diese Funktionen sind Callbackfunktionen, die aktiv von den Modulen und den NC-Workingsteps an den Eingriffsschnittstellen angemeldet werden. Sie registrieren dabei eigene Funktionen, die automatisch nach einer Änderung aufgerufen werden. Der Ansatz arbeitet somit schneller, da er von Nachrichtenpuffern und Pipelines unabhängig ist. Gemäß der Anforderungsanalyse sind so direkte Prozesseingriffe möglich, wie die Manipulation der Werkzeugzustellung oder der operationsspezifischen Vorschübe.

Verglichen mit dem Austausch von Nachrichten ist der Eventmechanismus schneller. Jedoch sind die Events auf explizite Parameter und NC-Workingstepobjekte angewiesen, bei denen sie sich registrieren können. Sie können nicht, wie die Nachrichtenpipes, umfangreiche NC-Workingsteps und NC-Workingsteplisten in einer definierten Abarbeitungsfolge austauschen.

6.4. Weg- und funktionssynchrone Umsetzung der NC-Workingsteps

Synchronizing the execution of the NC-workingsteps with toolpaths and NC-functions

Der konzipierte Interpreter setzt die im NC-Programm kodierten Arbeitsschritte und Bearbeitungsvorgaben in eine Liste steuerungsinterner NC-Workingstepobjekte um. Die NC-Workingstepobjekte enthalten jedoch, wie in Kapitel 6.3 erwähnt, noch keine expliziten Sollwerte und Schaltanweisungen. Sie können daher nicht unmittelbar von den Antriebsreglern und den Schnittstellen der NC-Steuerung ausgeführt werden. Zuvor müssen sie unter den kinematischen und funktionsablauftechnischen Vorgaben der Werkzeugmaschine weiter verarbeitet werden. Diese Aufgabe übernehmen die in Bild 6.1 dargestellten, dem Interpreter nachfolgenden Steuerungsmodule. In Abhängigkeit von dem verfügbaren freien Arbeitsspeicher der NC-Steuerung sowie dem Bearbeitungsfortschritt leitet der Interpreter ihnen über sein Frontend, d.h. die beschriebenen Kommunikationsmechanismen die einzelnen NC-Workingsteps zur Verarbeitung weiter. Die verarbeitenden Module lassen sich in zwei Funktionsbereiche gliedern: Die Geometrieverarbeitung zur Ansteuerung der Maschinenachsen und die synchrone Ansteuerung der Schaltfunktionen über die Speicherprogrammierbare Steuerung (SPS). Aufgrund ihrer unterschiedlichen steuerungstechnischen Arbeitsweise erfordern die beiden Teilbereiche angepasste Lösungskonzepte zur Realisierung laufzeitparalleler Programm- und Prozesseingriffe:

- Bei Manipulationen der dynamischen Werkzeug- und Achsbewegungen sind die kinematischen Grenzen und die Trägheit der Maschinenachsen zu berücksichtigen.
- Über die SPS sind Eingriffe aufgrund ihrer zyklischen Arbeitsweise und den Ablauffolgen nur zu diskreten Zeitpunkten und Funktionszuständen möglich.

6.4.1. Geometriekette

Geometry processing

Die Geometriekette umfasst alle steuerungsinternen Funktionen, die auf die Programmauflösung und Schnittaufteilung durch den Interpreter folgen und bis zur Bereitstellung von Sollwertfunktionen für die einzelnen Maschinenachsen abzuarbeiten sind. Hierzu zählen die Werkzeugkorrektur, die Verrechnung von Nullpunktverschiebungen, die kinematische Transformation, die Geschwindigkeitsführung und die Interpolation der Bahnkurven zu zeitdiskreten Sollpositionen und Beschleunigungen. Sie hat die Aufgabe, die in den NC-Workingsteps generierten Werkzeugbahnen an die Werkzeuggeometrie, den Werkzeugverschleiß, die Maschinenkinematik und die Geschwindigkeits- und Beschleunigungsgrenzen der Maschinen-

achsen anzupassen. Dazu werden mehrere 100 Bahnsegmente im Vorlauf eingelesen, ihre Übergänge innerhalb vorgegebener Toleranzen geglättet und, mit dem Ziel einer schnellen datentechnischen Weiterverarbeitung, zu möglichst langen Kurven zusammengefasst. Das Ergebnis sind Bewegungskurven, z.B. Spline, die sich mehrfach stetig differenzieren lassen, um sie ruckfrei über die Maschinenachsen abfahren zu können [FAUS97, SIEM04].

Die segmentübergreifende Arbeitsweise konventioneller Geometrieketten steht im Widerspruch zu der Idee, jede Abarbeitungssituation eindeutig einem NC-Workingstep zuzuordnen und über diese unmittelbar in die Abarbeitung eingreifen zu können. Die vorausschauende Bahn- und Geschwindigkeitsführung führt zu einer Datenpufferung und zu internen Abhängigkeiten zwischen den einzelnen Bahnsegmenten. Damit nimmt die Reaktionszeit zwischen einem Eingriff und der gewünschten Bahnadaption zu. Ferner geht aufgrund der Glättung und Komprimierung der Bewegungskurven der Bezug zu den ursprünglichen NC-Workingsteps sowie den darin enthaltenen Bahnsegmenten und Prozessinformationen verloren. Um schnell und kontextorientiert in den Prozess eingreifen zu können, muss daher der Vorlauf kontrolliert und der Bezug wiederhergestellt werden.

6.4.1.1. Kompensation des Vorlaufs nach einem Eingriff

Considering the look a-head after manipulating the programmed set-values

Es ist nicht möglich, jeden Parametereingriff, analog zur Spindeldrehzahl, unmittelbar über einen Offset mit den Sollwerten zu verrechnen. Beispielsweise ist die axiale Zustellung eines Werkzeugs nur bei Maschinen mit einer zur Werkzeugachse parallelen Maschinenachse direkt manipulierbar. Anderenfalls muss die Steuerung gezielt einen neuen Bahnverlauf berechnen, den Bearbeitungsfortlauf verfolgen und an einer geeigneten Stelle auf die adaptierte Bahn wechseln. Hierzu muss das entworfene Eingriffskonzept die weiter fortlaufenden Achsbewegungen vor dem Hintergrund der zur Adaption erforderlichen Verarbeitungszeit berücksichtigen. Die Rechengeschwindigkeit und der Vorlauf der Geometriekette sind daher kritische Größen innerhalb der NC-programmbasierten Prozessregelung. Sie müssen minimiert und ihr negativer Einfluss kompensiert werden.

Zur Kontrolle des Vorlaufs bietet sich der Ansatz an, die Teilfunktionen der Geometriekette zu erweitern. Kann ein Eingriff nicht unmittelbar durch Verrechnung von Offsets umgesetzt werden, wird in den verschiedenen Steuerungsmodulen eine Zwischenstrategie ausgelöst. In einem ersten Schritt meldet das den Eingriff verarbeitende Modul die Zeit zurück, innerhalb der die NC-Workingsteps adaptiert, neu generiert und zur Verarbeitung bereitgestellt werden. Ausgehend von dieser Zeit und dem aktuellen Bearbeitungsfortschritt schätzt die Geometriekette einen Synchronisationspunkt ab und liefert diesen mit seinen organisatorischen und technologischen Informationen an das die Adaption berechnende Modul zurück. Je nach implementiertem Kommunikationskonzept bieten sich hoch priore Nachrichten, Interrupt Funktionen oder Semaphoren an.

Der Synchronisationspunkt ist gleichzeitig der Endpunkt der alten, zu korrigierenden Bewegungsbahn und der Startpunkt der neu zu generierenden, adaptierten Bewegung. Aufgrund der Glättung innerhalb der Geometriekette, sind der Synchronisationspunkt und die Vorgaben der vorangehenden Module nicht deckungsgleich. Daher wird dieser Punkt toleranzbehaftet über die vorgestellte NC-Workingstepliste gesucht und als neuer Startpunkt ausgewertet.

Die Teilfunktionen der Geometriekette löschen alle dem Synchronisationspunkt nachfolgenden Elemente und leeren ihre Vorlaufpuffer. Die letzte gepufferte Vorgabe ist damit der Synchronisationspunkt. Über die leeren Vorlaufpuffer können die neu bereitgestellten, adaptierten Bahnsegmente direkt im Anschluss an den Synchronisationspunkt eingelesen werden. Die Geometriekette ist durch den Eingriff so parametriert, dass sie aus möglichst wenigen Bahnsegmenten die Werkzeug- und Geschwindigkeitskorrekturen sowie die Transformation berechnet. Die Puffer- und Rechenzeiten werden minimiert. Dem wird das Ziel langer, homoge-

ner Bahnsegmente mit maximaler Ausnutzung der Geschwindigkeitsprofile untergeordnet. So wird die Verarbeitungszeit verkürzt, um schnellstmöglich adaptierte Kurvenfunktionen bereitzustellen. Gesteuert über das Eintreffen der neuen Segmente wird nach der umgesetzten Prozesskorrektur der volle Funktionsumfang der Geometriekette wieder freigegeben.

Etwaige Beschleunigungssprünge im Synchronisationspunkt gleicht die Geschwindigkeitsführung aus. Sie greift hierzu rückwärts auf die verketteten NC-Workingsteplisten zu, liest die vorangehenden Geschwindigkeitsprofile aus und gleicht sie mit dem Ziel einer harmonischen Bewegungsführung mit den nachfolgenden, neuen Bahnprofilen ab. Dadurch, dass die NC-Workingsteps durchgängig korrigiert werden, stehen die adaptierten Vorgaben gleichzeitig auch nach außen zur Prozessüberwachung und Ablaufverfolgung zur Verfügung.

Der Interpolator stellt parallel sicher, dass der Synchronisationspunkt erst erreicht wird, wenn die adaptierten nachfolgenden Bahnen bereitstehen und vorverarbeitet sind. Dazu verzögert er die Abarbeitung. Im Fall der Fräsbearbeitung reduziert er fortlaufend die Bahngeschwindigkeit, indem er auf das Geschwindigkeitsprofil einen langsameren Interpolationstakt anwendet. Die verbleibende Zeit ermittelt er aus Statusmeldungen des Interpreters und der Kommunikationsplattform. Hinter dem Synchronisationspunkt hebt er dann den Takt linear wieder auf die Ausgangswerte an.

6.4.1.2. Bezug der expliziten Achsbewegungen zu den NC-Workingsteps

Linking the explicit axes movements to the NC-workingsteps

Aus dem Überschleifen und Zusammenfassen einzelner Bahnsegmente resultieren lange Bewegungsbahnen. Diese spiegeln nicht mehr zwingend die aufgaben- und operationsorientierte Segmentierung der NC-Workingsteps wider. Beispielsweise lässt sich im tangentialen Übergang zwischen der Anrückbewegung und der Bearbeitungsbahn nicht mehr unterscheiden, wo das Werkzeug frei steht und ab welchem Punkt es sich im Eingriff befindet. Die Eingriffsfunktionen lassen sich weniger präzise steuern.

Die Lösung besteht in einer gezielten Segmentierung innerhalb der Geometriekette sowie der Erweiterung der die Bewegungsbahnen beschreibenden Datenobjekte:

- Gezielte Segmentierung entsprechend den NC-Workingsteps
- Erweiterte Datenobjekte, um die Informationen mehrer NC-Workingsteps über die einzelnen Segmente zu parametrieren

Gezielte Segmentierung

Anstelle die Kurvenverläufe ausschließlich aufgrund der aus dem Interpolationstakt in Kombination mit den Beschleunigungsgrenzen resultierenden Bahnabweichungen zu segmentierten, werden zusätzliche Kriterien eingeführt. Hierzu stellen der Interpreter, bzw. die von ihm erzeugten NC-Workingstepobjekte zu jedem Bahnsegment aufgaben- und operationsorientiert Technologieinformationen bereit. In der Geometrieverarbeitung werden die Technologieinformationen erkannt und vor der Bahnverrechnung ausgewertet. Wenn eine Bewegungsbahn präzise einem NC-Workingstep entsprechen soll, so leitet die Geometrieverarbeitung eine gezielte Zwangssegmentierung ein.

Indem der Entwickler seine eigenen Routinen in die Geometrieverarbeitung integrieren kann, hat er die Möglichkeit, die NC-Workingsteps gezielt auszuwerten und den Bahnverlauf gemäß seinem Prozessmodell optimiert zu segmentieren und später gezielt in die Bearbeitung eingreifen zu können. Während er im freien Raum, wie beispielsweise bei Positionierbewegungen, ruckartige Übergänge aufgrund grober Segmentgrenzen zulässt, definiert er diese innerhalb einer Bearbeitungsbahn in Abhängigkeit von den Beschleunigungsprofilen, den zu erwartenden Bearbeitungskräften und Eingriffsbedingungen.

Erweiterte Datenobjekte

Da sich auch innerhalb eines Segments Parameter ändern oder dieses mehrere NC-Workingsteps überspannt, sieht das Konzept zusätzlich vor, dass die Technologieparameter und Kontextinformationen der NC-Workingsteps über den Verlauf der generierten Bahnen parametriert werden. Zu diesem Zweck werden die Datenobjekte, in denen die Bewegungssegmente datentechnisch verwaltet werden, um zusätzliche Attribute erweitert. Dieser Ansatz wird heute bereits verwendet, um die Werkzeugleistung oder die Werkzeugdrehzahl bahnsynchron zu steuern. Aufgrund der Zielstellung, eigene NC-Workingsteps und Bearbeitungsmodelle in die Steuerung integrieren zu können, werden jedoch die Attribute nicht starr vorgegeben. Stattdessen stellen die Segmentobjekte ihren Bahnverlauf als Parameter offen zur Verfügung. Der Entwickler hat so die Möglichkeit, seine eigenen Datenstrukturen positionsgenau zu verknüpfen. Denkbare Attribute sind die Art des Segmentübergangs und des Werkzeugeingriffs, die bei einer Prozessregelung zu berücksichtigenden Technologiegrenzwerte oder vordefinierte Eingriffstrategien (vgl. Kapitel 6.5).

Grenzwertüberprüfung nach einem Eingriff

Die Module der Geometriekette puffern mehrere Bahnsegmente, um diese übergreifend auszuwerten und optimierte Bewegungsprofile berechnen zu können. Das Ergebnis sind angepasste Vorschubgeschwindigkeiten und ruckfrei geglättete Bahnen. Streng genommen ist dieses Vorgehen der NC-Steuerung ein erster Eingriff in die NC-Programmvorgaben. Greift nun zusätzlich eine externe Regelung oder der Werker ein, so ist sicherzustellen, dass in Summe die zulässigen Grenzwerte nicht überschritten werden.

Der Ansatz, den Prozess in diskreten Schritten zu adaptieren, erlaubt es, die zuvor generierten Bewegungsbahnen in die vorgelagerten Module zurückzuführen, dort die aus der Verrechnung resultierenden Abweichungen zu bestimmen, zu bewerten und in der folgenden Prozesskorrektur zu berücksichtigen. Folglich stellen die adaptierenden Module, bzw. die darin implementierten NC-Workingstepfunktionen sicher, dass in Summe die vorgegebenen Grenzwerte eingehalten werden.

Werden dagegen Prozesskennwerte durch Offsets kontinuierlich geändert, wie die axiale Werkzeugzustellung oder der Bahnvorschub, so dürfen die Offsets nur unter Beachtung von technologischen Grenzwerten verrechnet werden. Für die Module der Geometriekette folgt daraus, dass sie nicht nur die Sollparameter sondern auch deren Grenzen durchgängig bahn- und ablaufsynchron verarbeiten und an den adaptierbaren Sollwertschnittstellen bereitstellen müssen (Bild 6.13). Innerhalb der NC-Workingsteps werden dazu neben den programmierten Sollvorgaben zusätzlich die im NC-Programm kodierten oder aus der steuerungstechnischen Programmumsetzung ermittelten Parametergrenzen gespeichert. Diese lösen sich in elementarere, die einzelnen Bewegungsbahnen beschreibende NC-Workingsteps auf. Dabei prüfen die Algorithmen zur Bahngenerierung, ob die Parameter zusätzlich einzugrenzen sind. Beispielsweise ließe sich beim Fräsen der maximale Vorschub in Abhängigkeit von der aus dem Krümmungsradius der Bahn resultierenden Werkzeugumschlingung parametrieren. Die Geometriekette muss diese Informationen auf das aktuelle Werkzeug, die Achskinematik und die Beschleunigungsgrenzen der Antriebe abbilden. Ausgehend von einer parametrierten Geometriekette, wird dazu jeder technologische Grenzwert als zusätzliche Funktion, z.B. ein Spline, über dem Verfahrweg oder der Bearbeitungsfortschritt parametriert. Das Ergebnis ist eine Funktionsschar, die neben den Achsbewegungen die verschiedenen Prozessparameter, wie minimale und maximal korrigierbare Vorschübe, Drehzahlen oder Hilfsvektoren, um die Werkzeugbahn in der laufenden Bearbeitung verschieben zu können.

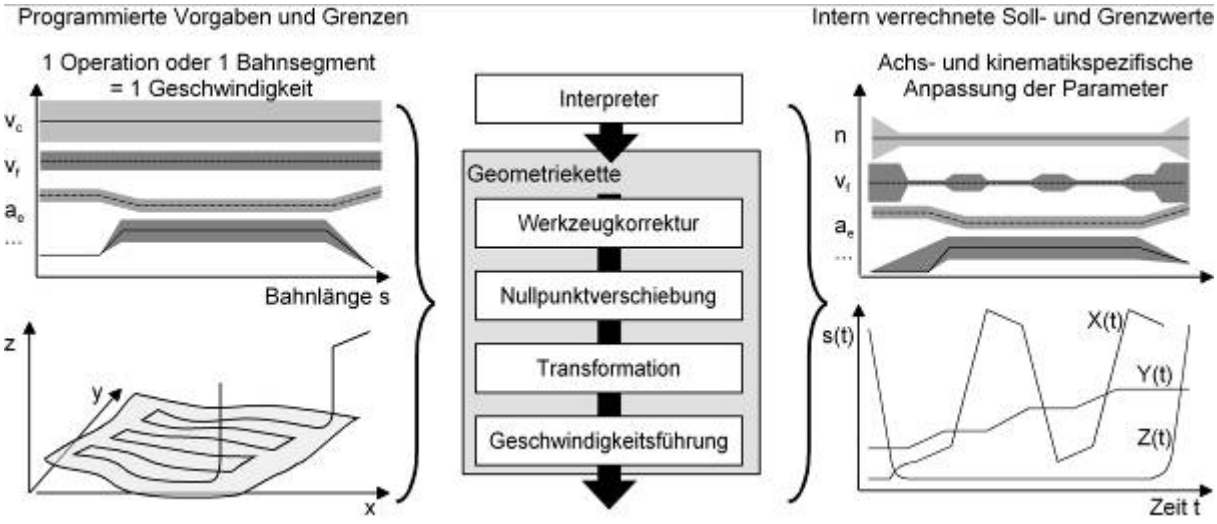


Bild 6.13: Erweiterte Geometriekette mit über die Bewegung parametrierten Grenzwerten
Extended NC-kernel in order to parameterize the limits based on the toolpaths

6.4.2. Synchronisation der Eingriffe zwischen NC und SPS

Synchronizing manipulations between the NC and the PLC

Über die Anpasssteuerung (SPS) werden die verschiedenen Maschinenfunktionen ereignisorientiert angesteuert. Sie arbeitet zyklisch vordefinierte Funktionsabläufe ab, überwacht zusammen mit der Programmausführung die aktuellen Maschinenzustände und verriegelt andere Funktionen situationsabhängig. Durch die Ablaufsteuerung wird sie fortlaufend mit dem Programmfortschritt und den aktuellen Achspositionen abgeglichen. Diese Synchronisationsmechanismen stellen sicher, dass ein Werkzeug erst nach Abschluss des Werkzeugwechsels zugestellt wird oder eine Schnittbewegung erst nach dem Anlauf der Arbeitsspindel beginnt.

Vor dem Hintergrund der konzipierten NC-Programmierschnittstelle und den steuerungsseitig daraus abgeleiteten NC-Workingsteps bieten sich diese Synchronisationsmechanismen an, um Arbeitsschritte, Operationen oder Abläufe mit gleichen Technologieparametern als zusammenhängende Prozess-Soll-Zustände zu interpretieren und die entwickelten Eingriffsoptionen gezielt zu steuern. Die Ablaufsteuerung wertet die beschriebenen Bahnsegmente der Geometriekette und die Funktionsroutinen des SPS aus, um Momente zu definieren, zu denen der Anwender oder eine Regelung die Abarbeitung gefahrlos und ohne negative Einflüsse auf das Bearbeitungsergebnis unterbrechen können (vgl. Kapitel 6.2.4).

Ferner steuert die Ablaufsteuerung die NC-technisch möglichen und prozesstechnisch zulässigen Eingriffe. Dazu wertet sie die aktuell anstehenden NC-Workingsteps und die innerhalb der auszuführenden Bahnsegmente und Schaltfunktionen parametrierten Zusatzinformationen aus. In Abhängigkeit von den darin enthaltenen Vorgaben werden steuerungsinterne Zustände gesetzt, die ihrerseits die nach außen aktiven Eingriffsschnittstellen freigeben, kontextabhängig vorparametrieren oder verriegeln. Für die im Rahmen dieser Arbeit entwickelte Betriebsart *„Prozessregelung über NC-Programmparameter“* sind in Bild 6.14 die unterschiedlichen Zustände in Form eines Zustandsautomaten dargestellt. Alle Steuerungsfunktionen warten auf das Erreichen der darin definierten statischen Zustände, bevor die Programmvorgaben adaptiert, mit der laufenden Bearbeitung synchronisiert, ausgeführt und final innerhalb der Überwachung auf ihre Wirksamkeit hin überprüft werden.

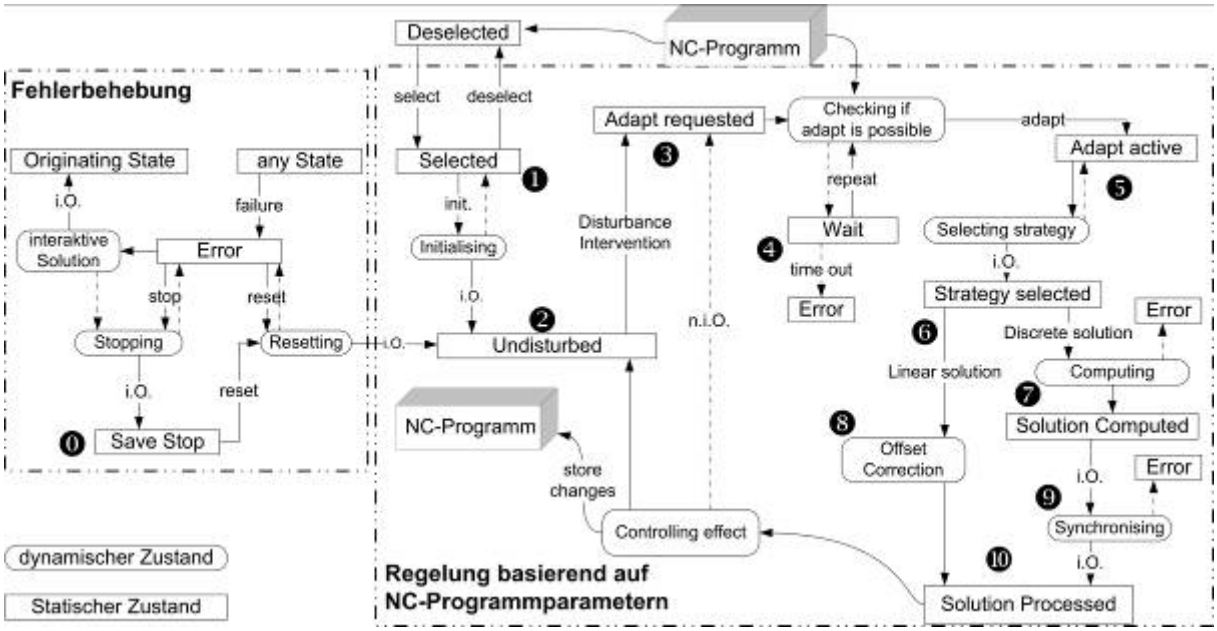


Bild 6.14: Beispiel für den Zustandsautomaten der Betriebsart „Prozessregelung“
State machine for the mode „process control“

Der Zustandsautomat wird über die Anwahl ❶ des NC-Programms aktiviert. Nachdem die NC-Steuerung ihre internen NC-Workingsteps initialisiert hat, wechselt sie in einen ungestörten, betriebsbereiten Zustand ❷. Ausgelöst durch einem Werker- oder Regelungseingriff wechselt die Steuerung in den Zustand „Adapt requested“ ❸. Dieser Zustand dient den Steuerungsmodulen und den darin gepufferten NC-Workingsteps als Trigger. Zielt der Eingriff erst auf einen nachfolgenden NC-Workingstep, so wechselt der Zustandsautomat vorrübergehend auf „Wait“ ❹. Parallel wird die Eingriffsinformation durch die NC-Workingstepliste durchgereicht, bis dass ein NC-Workingstep auf den Eingriff reagiert. Dieser initiiert den Zustand „Adapt active“ ❺ und beginnt seine Parameter und Sub-Workingsteps zu adaptieren. Die nächsten Zustände spiegeln wieder, dass eine Korrekturstrategie ausgewählt wurde ❻ und dass die internen Berechnungen neuer Parameter und Sub-Workingsteps abgeschlossen wurden ❼. Diese Zustandsänderung wird zeitlich überwacht, um bei Bedarf in einen Fehlerzustand zu wechseln und die Bearbeitung abzubrechen ❽. Ferner wird der Zustand in der Anzeige oder der Regelung zur Fortschrittsverfolgung ausgewertet. Danach werden je nach Strategie entweder die Sollwerte über Offsets manipuliert ❽ oder die Steuerung wechselt auf die neu errechneten, adaptierten Verfahrwege ❾. Sobald die Korrekturen über die Sollwertschnittstellen ausgeführt werden, meldet die Steuerung den Zustand „Solution Processed“ ❿. Aus diesem Zustand kann entweder die Korrektur fortgesetzt werden, um z.B. die Werkzeugzustellung schrittweise zu reduzieren, oder der Zustandsautomat wechselt zurück in seine Grundstellung ❷, um das jetzt adaptierte NC-Programm weiter abzuarbeiten.

Bei diskreten Regelschritten, wie sie bei der Neuberechnung von Bewegungsbahnen entstehen, sperren die Zustände sich gegenseitig und stellen sicher, dass der Prozess erst nach erfolgter Korrektur bewertet ❿ wird und bis dahin keine weiteren Regeleingriffe zugelassen werden. Für den Fall, dass eine Regelstrategie nicht in einer definierten Zeit zum Ziel führt, wird eine Notfallstrategie ausgelöst. Der Prozess wird unterbrochen, um mit dem Facharbeiter interaktiv das Problem zu lösen, oder er wird kontrolliert abgebrochen ❽.

Zur technischen Umsetzung bieten sich zwei Lösungsansätze an: Die Steuerung kann die Zustände zentral erfassen und als explizite Variablen nach außen offen legen. Alternativ halten die NC-Workingsteps ihren aktuellen Bearbeitungszustand eigenständig nach und erlauben

hierauf über externe Schnittstellen zuzugreifen. Vor dem Hintergrund einer offenen Erweiterung der Steuerungsfunktionen bietet eine dezentrale Verwaltung durch die NC-Workingsteps Vorteile. Die Zustände sind nicht auf steuerungsseitig vordefinierte Werte begrenzt, sondern können frei mit den NC-Workingsteps erweitert werden. Durch die Verkettung der NC-Workingsteps wird jede Zustandsänderung unmittelbar weitergemeldet und die programmierte Bearbeitungsvorgabe passt sich fortlaufend dem Abarbeitungsfortschritt an. Deshalb wird dieser Ansatz weiter verfolgt.

6.5. Umsetzung der Eingriffe und Übergangsstrategien

Computation of the manipulations and intermediate strategies

Ein Anwender oder eine Regelung haben ein vordefiniertes Prozessmodell und erwarten, dass ihre Eingriffe durch die NC-Steuerung analog zu diesem Modell schnell und nachvollziehbar umgesetzt werden. Dabei lassen sich unterschiedliche Ebenen unterscheiden, die sich durch den zeitlichen Vorlauf zwischen dem Eingriff und der steuerungstechnischen Ausführung unterscheiden:

- Eingriffe in die Reihenfolge der Arbeitsschritte
- Vorausschauende Optimierung von Prozessparametern
- Manipulation der aktuellen Prozessparameter
- Eingriffe in die Bahnverarbeitung

Im Folgenden werden für die vier Fälle optimierte Konzepte entwickelt und vorgestellt.

6.5.1. Änderung der programmierten Arbeitsschrittreihenfolge

Changing the sequence of coded STEP-NC workingsteps

Die im NC-Programm definierten Arbeitsschritte sind dadurch gekennzeichnet, dass sie sich mindestens durch einen der Parameter Werkzeug, Operation oder Bearbeitungsmerkmal unterscheiden. Zwischen zwei Arbeitsschritten ist daher das Werkzeug in der Regel nicht im Eingriff. Aus dieser Situation heraus kann die Bearbeitung gefahrlos gestoppt, umfangreich geändert und vor ihrer Fortsetzung auf Fehler und Kollisionen hin überprüft werden.

Neben den Arbeitsschritten bieten sich zur Ablaufmanipulation ferner die Arbeitspläne („*Workplan*“) und die als geschlossene Aufgaben auszuführenden NC-Funktionen („*Exchange Pallet*“, „*Load Tool*“, „*Unload Tool*“) an. Um darüber hinaus eigene Funktionen in die Programmierschnittstelle integrieren zu können und diese gezielt für Eingriffe freizugeben, werden die bereits im Rahmen der Programminterpretation vorgestellten Konfigurationsdateien und Schlüsselworte um Freigabeparameter erweitert. Nach dem Laden des NC-Programms vergleichen die NC-Steuerung sowie die Anwendungen der Benutzungsoberfläche den Programminhalt mit den Schlüsselworten und wenden die Freigaben an.

Wenn zudem die Wechselwirkungen und Abhängigkeiten der Arbeitsschritte und NC-Funktionen im Vorfeld überprüft werden, können sie dem Anwender grafisch angezeigt und kontextorientiert als Eingriffsoptionen angeboten werden. Mögliche Kriterien zur Auswertung sind die Wechsel des Werkzeugs, der Spannlage, der Strategie oder des zu bearbeitenden Merkmals. Des Weiteren lässt sich anhand der Merkmale und den an ihnen ausgeführten Operationen erkennen, inwieweit mehrere Arbeitsschritte aufeinander aufbauen und daher in ihrer Reihenfolge nicht verändert werden dürfen. Dieser Ansatz stellt beispielsweise sicher, dass ein Gewinde erst nach dem Bohren des Kernlochs geschnitten wird.

Änderung der ausstehenden Arbeitsschrittfolge

Der Interpreter bildet die im NC-Programm kodierten Arbeitsschritte auf seine steuerungsinternen NC-Workingsteps ab und verkettet sie entsprechend der im Arbeitsplan vorgegebenen Ausführungsreihenfolge. Greift der Anwender über die Benutzungsoberfläche in die programmierte Reihenfolge ein, so wird im Interpreter lediglich die Verkettung neu sortiert. Indem die NC-Workingsteps untereinander ihre End- und Startwerte abgleichen, werden fehlende Verbindungsbewegungen, notwendige Werkzeugwechsel und die zu ändernden Technologieparameter erkannt und autonom angeglichen. Komplexere Abläufe werden in Form neuer, für diese Aufgaben spezifischer NC-Workingsteps im Ablauf ergänzt.

Solange sich die Eingriffe auf die noch ausstehenden Arbeitsschritte beziehen, bleibt der NC-Steuerung ausreichend Zeit, um ihre internen Datenstrukturen zu adaptieren und den geänderten Arbeitsplan vorab in der Benutzungsoberfläche zur Anzeige zu bringen. Selbst wenn der Eingriff den unmittelbar anstehenden Arbeitsschritt betreffen würde, könnte die NC-Steuerung den Abarbeitungsfortschritt verlangsamen oder unterbrechen, um Zeit für die Adaption und das Generieren neuer NC-Workingsteps zu gewinnen. Diese Verzögerung wäre für den Anwender nachvollziehbar und sie hätte, abgesehen von der verlängerten Ausführungsdauer, keinen negativen Einfluss auf das Bearbeitungsergebnis.

Parallel zu den Eingriffen werten die NC-Steuerung und die in die Benutzungsoberfläche integrierten Simulations- und Programmierhilfen die geänderte Arbeitsschrittreihenfolge aus. Erkennen sie eine Kollision oder die Verletzung fertigungstechnisch notwendiger Prozessfolgen, so können sie die Bearbeitung vor dem nächsten Arbeitsschritt sicher anhalten, den Anwender informieren und auf seine Korrektur oder Bestätigung warten.

Direkter Wechsel ausgehend von einem laufenden Arbeitsschritt

Das Konzept erlaubt ebenso den Abbruch des aktuellen Arbeitsschritts, um auf einen anderen, nachfolgenden Arbeitsschritt zu wechseln. Jedoch reicht in diesem Fall das vorausschauende Ändern der Abfolge der generierten und in der Steuerung gespeicherten NC-Workingsteps nicht aus. Es muss zusätzlich der aktuelle Arbeitsschritt abgeschlossen oder seine Bearbeitung sicher abgebrochen werden.

In Kombination mit den Funktionen der Benutzungsoberfläche lassen sich dem Anwender die im Zusammenhang mit dem gewünschten Eingriff möglichen Optionen anzeigen. Somit ist es möglich, den aktuellen Arbeitsschritt sofort, erst am Ende eines Bahnsegments oder einer Zustellebene oder nach Abschluss der Operation abzubrechen. In Abhängigkeit von der gewählten Option werden wiederum die NC-Workingstepfunktionen autonom aktiv. Sie würden ausgehend von den übergeordneten, die Bearbeitungsaufgabe und die Operation beschreibenden NC-Workingsteps die zum Abbrechen zusätzlich erforderlichen Bahnen generieren, Maschinenfunktionen schalten und Synchronisationsmarken setzen. Entsprechend den Anforderungen befreit die Steuerung den Anwender von komplexen Aufgaben und unterstützt ihn durch situationsgerechte Abläufe optimal. Beim Freifahren des Werkzeugs stellt sie sicher, dass falsche Benutzervorgaben nicht zu Kollisionen führen.

6.5.2. Vorausschauende Prozessoptimierung

Provident process optimization

Zusätzlich zu den Eingriffen in die Abarbeitungsfolge fordert der Werker, die Operationsvorgaben im Prozess ändern zu können ohne dazu das NC-Programm abbrechen zu müssen. Hierzu könnte er die im NC-Programm enthaltenen Vorgaben an der Benutzeroberfläche dialoggeführt ändern und zur Umsetzung an die Steuerung übertragen. Mögliche Anwendungsfälle sind zusätzliche Werkzeugwechsel oder die Anwahl einer tangentialen Zustellbewegung vor der nächsten Bearbeitungsoperation.

Um die Eingriffsmöglichkeiten des Werkers zu erweitern bietet es sich ferner an, zusätzlich zu den Programmvorgaben auch die steuerungsinternen NC-Workingsteps als Eingriffsschnittstellen offenzulegen, Bild 6.15. Dadurch wird die grobe, arbeitsschrittbezogene Struktur des NC-Programms feiner untergliedert. Dieser Ansatz ermöglicht es, Prozessparameter gezielt für einzelne Zustellebenen zu optimieren, anstelle sie pauschal für das gesamte, abzuarbeitende Merkmal ändern zu müssen. Zudem sind die NC-Workingsteps ausgehend von der vorgestellten Segmentierung und Parametrierung über die Geometriekette und die SPS bis hin zur Ausführung durchgängig verfügbar. Der direkte Zugriff auf einzelne NC-Workingsteps und ihre Sub-Workingsteplisten, ermöglicht somit den direkten Eingriff in technologisch zusammenhängende Prozessabschnitte.

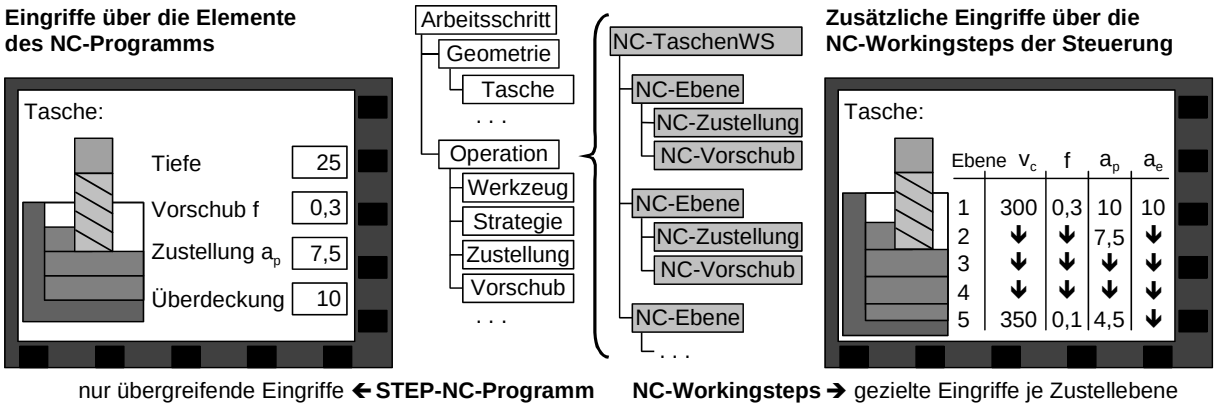


Bild 6.15: Prozesseingriff über die STEP-NC-Programmvorgaben und die NC-Workingsteps
Process manipulations based on the STEP-NC-program and the NC-Workingsteps

Aufgrund der entwickelten Lösung ist der Umfang der Eingriffsoptionen nicht allein auf die im NC-Programm kodierten Programmelemente und deren Parameter beschränkt. Über die Funktionen der von den Steuerungsherstellern oder externen Entwicklern in die NC-Steuerung integrierten NC-Workingsteps werden zusätzliche Funktionen realisiert und zur bearbeitungsparallelen Prozessmanipulation angeboten. Die Parameter und Zustände der NC-Workingsteps lassen sich zudem zur Fortschrittsverfolgung durch Überwachungssysteme oder die Benutzungsoberfläche auswerten.

Ferner erlaubt das Konzept die übergreifende Auswertung der NC-Workingsteps. Dabei werden einheitlich gültige Prozessparameter und gleiche, sich wiederholende Operationen erkannt. Indem dem Anwender diese Gemeinsamkeiten angezeigt werden, kann er die vorgenommenen Eingriffe gezielt auf die noch ausstehenden Arbeitsschritte übertragen. Erprobte Änderungen werden als Erfahrungswissen ohne eine Programmunterbrechung und Neuprogrammierung direkt in der Folgebearbeitung übernommen. Denkbare Anwendungen sind die programmweite Optimierung der Vorschübe im Eilgang, beim Zustellen und Freifahren sowie die Werkzeugzustellung oder die Schnittaufteilung. Die zur werkergerechten Anwendung erforderlichen Interaktionsmechanismen werden in Kapitel 7.2.3 entwickelt.

Ausgehend von der hierarchischen Verkettung der NC-Workingsteps lassen sich darüber hinaus komplexe Überwachungs- und Regelsysteme umsetzen. Dazu werden die NC-Workingsteps zur Systemparametrierung ausgewertet. In Abhängigkeit von einer erkannten Prozessabweichung kann dann gezielt in die elementaren Bewegungsbahnen und Technologieparameter oder aber pauschal in die Parameter zusammenhängender Sub-Workingsteplisten eingegriffen werden. Dieser Ansatz liefert die Grundlage, um Konzepte einer bahnsynchronen, über den Bearbeitungsprozess gezielt gesteuerten Schnittkraftüberwachung oder einer Rattervermeidung durch die steuerungsinterne Optimierung der Schnittaufteilung und Schnittparameter zu realisieren.

6.5.3. Direkte Eingriffe in die Sollwerte

Direct manipulation of generated set-values

Zusätzlich zu der vorausschauenden, diskreten Optimierung programmierter Ablauf- und Operationsvorgaben besteht im Bearbeitungsprozess die Notwendigkeit, einzelne Prozessparameter durch einen regelnden, d.h. abgleichenden Eingriff zu korrigieren und zu optimieren. Hierzu sind die vorgeschlagenen Mechanismen, bei denen die NC-Programmelemente oder NC-Workingsteps textbasiert in der Benutzeroberfläche korrigiert werden, nicht geeignet. Vielmehr müssen sich die Prozesskennwerte kontinuierlich manipulieren lassen. Dabei wird eine schnelle Umsetzung nicht nur aufgrund der hohen Prozessgeschwindigkeiten vorausgesetzt. Zudem erwartet der Anwender eine schnelle Rückmeldung, um seinen Eingriff bewerten und ggf. korrigieren zu können.

Generell ist die schnellste Methode, um in einen elektronisch gesteuerten Prozess einzugreifen, die Manipulation seiner auszuführenden Sollwerte. Sie werden anschließend über die I/O-Schnittstellen der Anpasssteuerung oder die Lagereglern der Antriebe ausgeführt. Jedoch umgeht der direkte Zugriff auf die Sollwerte den zuvor entwickelten Ansatz einer Regelung über die NC-Programmelemente und die daraus abgeleiteten, sich interaktiv anpassenden NC-Workingsteps. Optimierungen ließen sich nicht unmittelbar in das NC-Programm und zukünftige Bearbeitungen zurückmelden. Ferner müsste der Werker auf indirekte Steuerungs- und Maschinenparameter zugreifen, wie z.B. die Spindeldrehzahl, anstelle direkt die ihm aus der Planung bekannten, technologischen Parameter zu korrigieren, d.h. in diesem Beispiel die Schnittgeschwindigkeit.

Lösen lässt sich der Konflikt, indem Änderungen an den Sollwerten direkt an die NC-Workingsteps zurückgemeldet oder von diesen durch eine kontinuierliche Überwachung erkannt werden. Dabei wird der bisherige Ansatz umgekehrt, nach dem die Eingriffe von der allgemeinen, übergeordneten Aufgaben- oder Operationsbeschreibung hin zur elementaren Weg- und Schaltfunktion weitergeleitet werden. Beibehalten wird dagegen der interaktive Abgleich zwischen den einzelnen NC-Workingsteps. Letztlich können die Sollwerte als elementarste Einheit der in der Steuerung gespeicherten Sub-Workingsteplisten verstanden werden. Als solche bilden sie nur noch einen einzelnen, steuerungstechnisch ausführbaren Wert ab. Von anderen Sollwerten hängen sie aufgrund ihrer Reihenfolge ab, über die ihre zeit- und wegsynchrone Ausführung gesteuert wird.

Einen Eingriff in die Sollwerte erkennen die übergeordneten NC-Workingsteps oder parallel zu den Sollwertschnittstellen arbeitende Eingriffsfunktionen. Sie werten den Eingriff aus, prüfen seine Anwendbarkeit und leiten die Manipulation weiter. Mit dem Ziel schnell reagieren zu können, bietet es sich an, das Weiterleiten in zwei Ebenen auszuführen. Die erste Ebene greift unmittelbar auf die nachfolgenden, zwischengespeicherten Sollwerte zu und manipuliert diese analog zu dem erkannten Eingriff. Diese Lösung entspricht den heutigen Prozessmanipulationen durch Verrechnung von Offsets mit den von der Steuerung nach Programmvorgabe generierten Sollwerten. Parallel wird die Änderung nach oben, d.h. in die übergeordneten NC-Workingsteps gemeldet. Dort werden die Eingriffe im Kontext der Bearbeitungsaufgabe und Operationsbeschreibung durch die Prozesslogik der NC-Workingsteps ausgewertet und mit vorgegebenen Grenzwerten verglichen. Bei Grenzwertüberschreitungen sendet die Steuerung eine Fehlermeldung an die eingreifende Stelle, d.h. an eine Regelung oder die dem Anwender zur Interaktion dienende Benutzeroberfläche, und löst über die NC-Workingsteps vorimplementierte Notfallstrategien aus. Diese Strategien können vom Ignorieren des Eingriffs, z.B. bei einer über die Antriebsleistung hinausgehende Drehzahlsteigerung, bis hin zum Abbruch der Bearbeitung vor dem Überschreiten von Grenzlasten reichen. Ist der Eingriff dagegen technisch umsetzbar, so adaptiert der NC-Workingstep seine internen Parameter, korrigiert die von ihm generierten Sub-Workingsteps und meldet die Änderung an die ihm nachfolgenden und übergeordneten NC-Workingsteps weiter. Um Grenzwertüberschreitungen

in den Sub-Workingsteps zu erkennen, wird von den übergeordneten Instanzen die Umsetzbarkeit nach unten angefragt, bevor sie sich und alle Sub-Workingsteps adaptieren. Zusätzlich melden die NC-Workingsteps Fehler, wenn sie eine Adaption nicht umsetzen können. Die Fehlermeldung wird an die vorangehenden und die nächst höheren NC-Workingsteps zurückgemeldet, um dort die Adaption zu korrigieren oder abzubrechen.

Korrektur der programmierten, nachfolgenden Bahn- und Prozessführung

Dieser Lösungsansatz stellt sicher, dass die Parametereingriffe unmittelbar überprüft, ausgeführt und nachhaltig in die Programmumsetzung übernommen werden. Das Erreichen bestehender Grenzen wird frühzeitig erkannt und, indem die NC-Workingsteps die nachfolgende Bearbeitung adaptieren oder neu umsetzen, wird der Prozess so angepasst, dass neue, erweiterte Grenzen entstehen. Damit wird der Anforderung Rechnung getragen, den Bearbeitungsprozess nicht nur lokal zu korrigieren, sondern ohne Programmunterbrechung umplanen zu können. Beispielsweise lässt sich innerhalb der NC-Workingsteps auswerten, wie weit eine kartesische Werkzeugachse zurückgezogen oder zusätzlich zugestellt wurde. Über die Geometrie des Merkmals und die effektive Wirklänge des Werkzeugs errechnet der für die Bahnauflösung des Merkmals zuständige NC-Workingstep, ob zusätzliche Bearbeitungsebenen eingefügt werden müssen oder die finale Sollkontur überschritten wird.

Steuerung der Eingriffe in Abhängigkeit von der Bearbeitungsoperation

Die NC-Workingsteplisten fassen Teilaufgaben einer Bearbeitungsaufgabe strukturiert zusammen. Ihre Struktur ermöglicht den gezielten Zugriff auf eine Positionier-, Zustell- oder Bearbeitungsbewegung. Folglich lassen sich auch die zu manipulierenden Sollwerte über die ihnen übergeordneten Bahnsegmente auf diese, im Verständnis des Werkers technologisch zusammenhängenden Operationen und Funktionsabläufen zurückführen. Die Auswertung der übergeordneten Vorgaben in den Eingriffsschnittstellen erlaubt eine kontextorientierte Parametrierung und auf einzelne Operationen beschränkte Ausführung der Prozessmanipulation.

Das Einschränken von Eingriffen auf Prozessabschnitte wird bereits heute auf der Basis des verbreiteten G-Codes mit im Programmablauf ergänzten, systemspezifischen Zusatzkodes oder mithilfe einer globalen Unterscheidung zwischen Eilgang- und Vorschubbewegungen realisiert. Steuerungs- und Regelungshersteller, wie FANUC oder ARTIS, bieten entsprechend aufgebaute Systeme zur Vorschuboptimierung an [FANU04a, ARTI03]. Jedoch stehen die abschnittsweise oder steuerungstechnisch definierten Begrenzungen nicht in direktem Zusammenhang zu dem aufgaben- und operationsorientierten Denken der Planer und Werker.

Diese Einschränkungen werden umgangen, indem der Werker bei der Prozessmanipulation gezielt über die im NC-Programm kodierten Operationen und die daraus abgeleiteten NC-Workingsteps geführt wird. Dazu ergänzt der Interpreter jedes intern generierte Befehlsdatum, bzw. Wegsegment um zusätzliche technologische und ablauforganisatorische Informationen. Diese werden von der Ablaufsteuerung sowie der Geometriekette zeit- und wegsynchron weiterverarbeitet. Die I/O-Schnittstellen, der Interpolator oder die Achsregelschnittstellen werten die Zusatzdaten aus und kontrollieren darüber ob und wenn ja welche der Eingriffsfunktionen zur Anwendung kommen. In der Benutzeroberfläche und der Steuerungstafel werden parallel die Anzeigen, Schalter und Override in Abhängigkeit von den anstehenden Operationen vorbelegt (vgl. Kapitel 7.1.2). Dieser Ansatz erlaubt es dem Facharbeiter oder einer Regelung, gezielt Zustell- oder Schnittbewegungen abzubremsen, während beispielsweise Bewegungen ohne Werkzeugeingriff unverändert oder beschleunigt ausgeführt werden.

Indem die erweiterten Override für Vorschub oder Drehzahl gezielt auf technologisch zusammengehörige Segmente angewandt werden, kann die Steuerung im Fall der Fräsbearbeitung anstelle einer prozentualen Änderung direkt den programmierten Bahnvorschub v_f, den Vorschub pro Zahn f_z oder die Schnittgeschwindigkeit v_c auswerten und zur Anzeige bringen. Der Anwender greift nicht mehr nur relativ in einen Zahlenwert ein, sondern kann in den

technologisch relevanten Größen denken und regeln. Zusätzlich überprüft und begrenzt die Steuerung die Vorgaben: Für die Drehzahl gilt, dass sie während eines Werkzeugeingriffs nicht unter eine Mindestschnittgeschwindigkeit fällt. Der Bahnvorschub richtet sich nach den kinematischen Grenzen der einzelnen Achsen.

Darüber hinaus werden die konventionellen Override diesen Funktionen überlagert. Dieses Konzept ist notwendig, um dem Facharbeiter weiterhin die ihm bekannten, übergreifend wirkenden Eingriffsmöglichkeiten anzubieten. Somit lassen sich auch NC-Workingsteps, in denen keine spezifischen Eingriff- oder Notfallstrategien implementiert wurden, extern manipulieren.

Rückmeldung der Eingriffe zur Optimierung zukünftiger Bearbeitungen

Ein weiterer Vorteil dieses Konzeptes besteht in der Möglichkeit, die in den NC-Workingsteps generierten Verfahrwege und Schaltabläufe sowie die nach Eingriffen adaptierten Parameter in Form eines neuen NC-Programms zurückzumelden. Auf dieser Grundlage kann der Werker den realen Ablauf im WOP-System simulieren und entscheiden, ob er die Optimierungen in die nachfolgenden Bearbeitungsprogramme überträgt. Dabei wird er von der objektorientierten Struktur des NC-Programms unterstützt. Sie erlaubt es ihm, Operationen, Strategien, Technologieparameter, Werkzeugdaten oder Verfahrwege als zusammenhängende Einheiten zu übernehmen und Arbeitsschritten zuzuweisen.

Ferner erlaubt die Rückmeldung, die Optimierung zwischen der geplanten und der real ausgeführten Bearbeitung mithilfe von Wissenssystemen in der Planung zu analysieren. Diese Aufgabe ist jedoch sehr komplex. Während der Werker selber den Prozess verfolgt und aufgrund seiner Beobachtungen in diesen eingegriffen hat, fehlt in der Planung das Wissen über die im Prozess aufgetretenen Kräfte, Vibrationen und Störungen. Um diese Auswertung leisten zu können, sind neben Algorithmen, die den Prozess bewerten, Datenelemente zum Prozessmonitoring zu entwickeln, die sich über die Bearbeitungsbahn und den Programmablauf parametrieren lassen. Ausgehend von der Leistungsfähigkeit der entwickelten Programmierschnittstelle sind verschiedenen Forschungsvorhaben entstanden, die sich mit der Definition von Elementen zur Protokollierung befassen und aus dem zurückgemeldeten Ablauf Erfahrungswissen für Neuplanungen extrahieren [KLEM05, BREC06, EXAP07].

6.5.4. Eingriffe in die Bahnverarbeitung

Manipulating the toolpath computation

Aufgrund der Maschinenkinematik und der zu fertigenden Merkmalsgeometrie sind insbesondere Eingriffen in die Bahnführung Grenzen gesetzt. Zum einen können die Parameter Zustellung und Eingriffsbreite nur innerhalb von Bandbreiten variiert werden, um das Werkzeug nicht zu überlasten und um Kollisionen zu vermeiden. Zum anderen entstehen durch die Eingriffe neue Konturen und Absätze am Werkstück, Bild 6.16. Diese führen zu ungleichmäßigen Zerspanungsquerschnitten und damit zu wechselnden Schnittkräften. Die adaptive Bahngenerierung muss daher das verbleibende Materialvolumen intelligent abarbeiten.

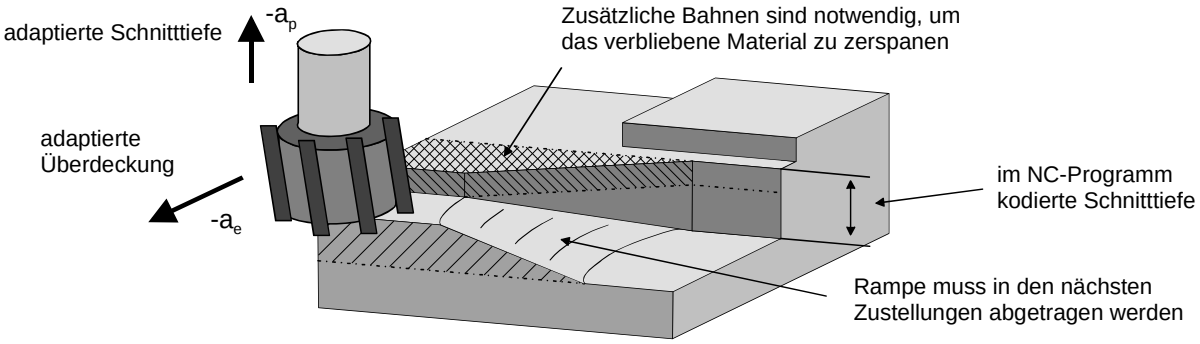


Bild 6.16: Einfluss der Eingriffe auf das Zerspanvolumen und die folgende Bahnauflösung
Effect of a manipulation with regard to the removal volume and the next toolpaths

Diese Aufgabenstellung wird in den der auf die Abarbeitung der unterschiedlichen Bearbeitungsmerkmale spezialisierten NC-Workingsteps gelöst. Ausgehend von den vorgegebenen Strategien, Technologieparametern und Werkzeugdimensionen lösen ihre internen Algorithmen das Bearbeitungsvolumen auf. Ferner werden die einzelnen Zustellebenen in eigenständigen NC-Workingsteps zusammengefasst, um diese parallel vorzuverarbeiten und im Prozess gezielt adaptieren zu können. Dabei werden die im NC-Programm kodierten technologischen und geometrischen Grenzen als auch die Randbedingungen der Maschine überprüft. Grenzwertüberschreitungen meldet die Steuerung als Fehler an die Benutzungsoberfläche und gibt dem Werker die Möglichkeit die Vorgaben im Prozess zu korrigieren, ohne das Programm unterbrechen zu müssen.

Während der Bearbeitung aktualisieren die NC-Workingsteps fortlaufend ihr internes Berechnungsmodell. Dazu werten sie die Statusmeldungen ihrer Sub-Workingsteps aus und rufen die aktuellen, kartesischen Positionswerte aus der Geometriekette ab. Die bereits abgearbeiteten Bahnen können beispielsweise als freier Raum für Rückzugs- und Zustellbewegungen interpretiert werden. Ihre Hüllkurven werden im Fall einer Taschenbearbeitung als Inseln behandelt, so dass nach einem Eingriff nur noch das unbearbeitete Volumen innerhalb der adaptierten Bahnauflösung verrechnet werden muss. Unnötige, zeitintensive Luftschnitte werden vermieden. Ausgehend von der Ist-Kontur werden Bahnen mit gleichmäßigen Zerspanquerschnitten generiert.

6.5.4.1. Diskrete Korrektur der Bahnführung

Discrete toolpath correction

Nicht jede Schnittaufteilung lässt sich vollvariabel manipulieren. Beispielsweise werden bei der Aufteilung einer Taschengeometrie in definiertem Abstand zur Fertigteilkontur parallele Hilfsbahnen, die sogenannten Äquidistanten berechnet. Diese müssen miteinander geschnitten und in spitzen Ecken oder Kreisbögen zusätzlich verkürzt werden, um die Sollgeometrie nicht zu verletzen und das Volumen in Abhängigkeit vom Werkzeugdurchmesser vollständig zu zerspanen. Nach jeder Änderung des Werkzeugdurchmessers und der Werkzeugüberdeckung müssen die Äquidistanten neu berechnet werden. Eine kontinuierliche, lotrechte Verschiebung der einzelnen Bahnen würde insbesondere in den genannten Ecken zu Kollisionen führen.

Die Lösung besteht in einer Kopplung zwischen der Regelung auf Sollwertebene und der Bahnauflösung des Interpreters. Ausgehend von dem aktuellen Wegsegment und der darauf erreichten Achsposition fragt die Regelung beim Interpreter einen möglichen Synchronisationspunkt für den Übergang auf eine adaptierte Bahn an. Der Interpreter schätzt die erforderliche Zeit bis zur Bereitstellung der adaptierten Bahnen ab und bestimmt ausgehend von den

ursprünglich generierten Bahnen, der Verfahrgeschwindigkeit und den Korrekturgrenzen einen Synchronisationspunkt. Die Bestimmung des Punktes ist besonders schnell, da der Interpreter die alten Weg-, Schalt- und Korrekturanweisungen in einer verketten Liste zwischengespeichert hat und damit zeitintensive Berechnungen einspart. Der Einsprungpunkt in die Liste wird zudem fortlaufend mit dem Abarbeitungsstand an der Antriebsschnittstelle aktualisiert. Nur Elemente, die abgearbeitet und nicht mehr relevant sind, werden gelöscht. Die Totzeit bis zur Rückmeldung des Synchronisationspunktes hängt daher im Wesentlichen von der Dauer für die Kommunikation zwischen der Regelung auf Sollwertebene und dem Interpreter ab.

Parallel zu den Anfragen nach einem Synchronisationspunkt und einer adaptierten Bahn greift die Regelung in die aktuellen Sollwerte ein. In Abhängigkeit von der an das Wegsegment gekoppelten Regelstrategie passt sie einzeln oder in Kombination die Vorschubgeschwindigkeit, die Spindeldrehzahl, die Zustellung und die Eingriffsbreite an. Den zurückgemeldeten Synchronisationspunkt wertet die Regelung als Zielpunkt aus, den sie innerhalb der Totzeit über die Regelgrößen ansteuert. Bis zu diesem Punkt werden alle Sollpositionen in einem Zwischenspeicher gepuffert und die darüber hinausgehenden, nachfolgenden Segmente gelöscht. Die Datenpuffer sind damit bereit, um die neue, vom Interpreter bereitzustellende adaptierte Bahn sofort aufzunehmen und im Anschluss an das Erreichen des Synchronisationspunktes auszuführen.

Reicht die Totzeit nicht aus, um die adaptierten Bahnen bereitzustellen, reduziert die Sollwertebene den Vorschub, um Zeit zu gewinnen. Im Grenzfall kommt die Bearbeitung zum Stillstand oder wird nach dem vollständigen Zurückziehen des Werkzeugs abgebrochen.

6.5.4.2. Kontinuierliche Korrektur der Bahnführung

Continuous toolpath correction

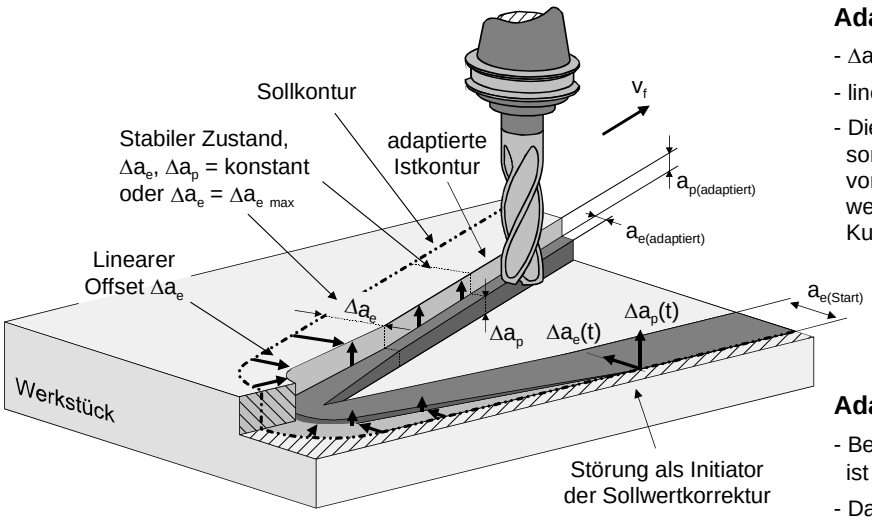
Andere Parameter lassen sich linear manipulieren. Hierzu zählen beispielsweise die Vorschub- und Schnittgeschwindigkeit oder, bei einer kartesischen Achsanordnung, die axiale Zustellung des Werkzeugs. Diese Parameter zeichnen sich dadurch aus, dass sie in der Regel direkt durch die Verrechnung von Offsets auf die auszuführenden Sollwerte adaptiert werden können. Dadurch entfallen das Vorausberechnen eines Synchronisationspunktes sowie die bis zu dessen Erreichen anzuwendenden Hilfsstrategien.

Während in der NC-Steuerung die Sollwerte linear adaptiert werden, halten die mit den Sollwerten verknüpften Wegsegmente und NC-Workingsteps die Eingriffe kontinuierlich nach. Über die Verkettung innerhalb der NC-Workingstepliste werden die Parameter der noch nicht ausgeführten, nachfolgenden sowie die der übergeordneten NC-Workingsteps automatisch aktualisiert. Diese werten die Eingriffe aus, um frühzeitig zu erkennen, ob die weitere Bearbeitung aufgrund von Konturverletzungen oder Grenzwertüberschreitungen abgebrochen werden muss und informieren den Anwender oder eine Regelung. Sobald sich der Prozess stabilisiert hat oder die externen Eingriffe stoppen, werden die Offsets an die NC-Workingsteps als zu übernehmende Korrektur gemeldet.

Über die Lagesollwerte ist eine Korrektur des Bahnverlaufs nicht immer möglich. Das größte Problem besteht darin, dass aus den auf die Maschinenkinematik transformierten Sollwerten nicht ersichtlich ist, in welche Richtung kollisionsfrei und technologisch sinnvoll korrigiert werden kann. Selbst die Werkzeugachse ist nicht immer eindeutig. Bei 5-Achskinematiken wird sie auf mehrere Maschinenachsen abgebildet. Die axiale Zustellung muss dann vor der Transformation, d.h. vor den steuerungsinternen Datenpuffern und folglich zeitlich entkoppelt zur Sollwertausführung erfolgen. Ein weiteres Defizit sind die fehlenden Informationen in rein wegorientierten NC-Programmen, wie z.B. nach DIN 66025. Weder Hinterschnitte noch die relative Lage zwischen dem Werkzeug und dem zu bearbeitendem Material sind kodiert. Entwicklungsansätze, wie der von *Versch* [VERS04], in denen alternative Bahnen anhand von Variablen im NC-Programm programmiert werden, sind suboptimal. Sie erfordern einen er-

höhten Programmieraufwand, bei dem alternative Verfahrwege vorgedacht werden müssen. Darüber hinaus kann eine Regelung die Variablen nur Satzweise, zu Beginn eines neuen Wegsegments korrigieren. Die Reaktionszeit auf eine Regelvorgabe würde mit der Weglänge, der Satzwechselzeit und dem Satzvorlauf variieren.

Auf der Sollwertebene lassen sich diese Einschränkungen nur durch zusätzliche Richtungs- und Bahninformationen lösen, entlang denen eine Regelung Offset-Werte errechnen und zu den ursprünglichen Lagesollwerten addieren kann. Die Informationen werden von den erweiterten Funktionen des Interpreters generiert und innerhalb der Geometriekette auf die Maschinenkinematik abgebildet sowie mit der Sollbahn parametriert (vgl. Kapitel 6.2). Sie enthalten die Wirkrichtungen der Zustelltiefe a_p und der Eingriffsbreite a_e bezogen auf die Maschinenachsen, Bild 6.17. Im Fall einer hinterschnittfreien Bearbeitung vereinfacht sich die Funktion zur Korrektur der Zustelltiefe auf einen Richtungsvektor. Für die Eingriffsbreitenadaption reicht eine Richtungsbeschreibung nicht aus. Insbesondere in Ecken und schmalen Nuten darf das Werkzeug nur so weit versetzt werden, dass es an benachbarten Konturen noch nicht zum Eingriff kommt und Konturverletzungen ausgeschlossen werden. Deswegen wird die Adaptionsgrenze der Eingriffsbreite in Form einer zweiten Bahn dargestellt, die mit der Sollkurve positionsbezogen verknüpft ist. Die Verknüpfung erfolgt je nach interner Darstellungsweise der Steuerung als Funktion über die Weglänge oder die Zeit. In der Gegenrichtung ist die maximale Eingriffsbreite in einer weiteren Kurve parametriert. Selbiges gilt für die maximale Zustelltiefe. Somit kann auf der Ebene der Sollwertverarbeitung auch eine Grenzwertregelung zur Prozessoptimierung realisiert werden.



Adaption der Eingriffsbreite a_e:

- Δa_e durch Folgebahnen begrenzt
- lineare Adaption bis $\Delta a_{e\ max}$
- Die Korrektur ist nicht als Richtung, sondern als 2. Bahn mit $\Delta a_{e\ max}$ vorgegeben. Die adaptieten Sollwerte werden zwischen den beiden Kurven $a_{e\ Soll}$ und $a_{e\ max}$ interpolliert.

Adaption der Schnitttiefe a_p:

- Bei $2^1/_2$D ohne Hinterschnitte ist Δa_p nicht begrenzt.
- Das Werkzeug kann vollständig zurückgezogen werden: $\Delta a_p = a_{p\ Start}$

Bild 6.17: Hilfsvektoren zur Sollwertkorrektur mit nachgeschalteter Bahnadaption

Auxiliary vectors to correct the set-values before restarting the toolpath generation

Der Vorteil dieses Lösungskonzepts liegt darin, dass alle geometrischen Berechnungen zeitunkritisch im Vorfeld der Achsregelung erfolgt sind. Der Regelung werden Korrekturwertfunktionen in Form von kollisionsfreien, beschleunigungs- und geschwindigkeitsbegrenzten Kurven in Maschinenkoordinaten bereitgestellt. Die Regelfunktionalität ist somit unabhängig von der Maschinenkinematik. Sie wertet die Kurvenfunktionen achsbezogen aus und verrechnet die Werte mit den Lagesollwerten.

Im Hintergrund wird die diskrete Bahnadaption aktiviert. Sie schätzt ab, wann die kontinuierliche Manipulation der Sollwerte ihre Grenzwerte erreicht und berechnet von diesem, in der Zukunft liegenden Punkt aus eine neue Sollbahn mit optionalen Korrekturvektoren. Zudem

wird die Bahnauflösung aufgerufen, sobald sich der Prozess stabilisiert hat, so dass optimierte Bahnen berechnet werden können, um das verbleibende Bearbeitungsvolumen zu zerspanen.

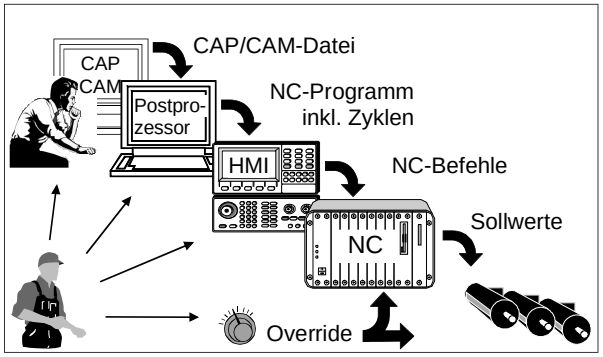
7. Benutzerschnittstelle zur NC-Programmierung und Manipulation

User interface for NC-programming and Manipulation

Die Untersuchungen von *Weydandt et al.* zur Anwenderinteraktion beim CNC gesteuerten Drehen und Fräsen haben aufgezeigt, dass in der konventionellen NC-Technik das automatisierte Abarbeiten eines Fertigungsprogramms im Vordergrund steht, während die Eingriffsmöglichkeiten des Werkers möglichst gering gehalten werden [WEYD99]. Steuerungsseitig sind daher nur begrenzte Regelparameter und Eingriffsfunktionen verfügbar. Hierzu zählen die Overrides für Vorschub und Spindeldrehzahl oder die Unterbrechungsfunktionen im Einzelsatzbetrieb [DAXL04]. Andere Parameter, wie die im NC-Programm kodierten Vorgaben, kann der Werker in der Benutzeroberfläche nur satzweise verfolgen, ohne sie im laufenden Prozess korrigieren oder optimieren zu können [KIEF05]. Automatisierte Überwachungs- und Regelsysteme sind ebenfalls auf Eingriffe in die Steuerungsparameter und Sollwerte begrenzt, abgesehen von spezialisierten Lösungen mit Steuerungsvariablen im NC-Programm.

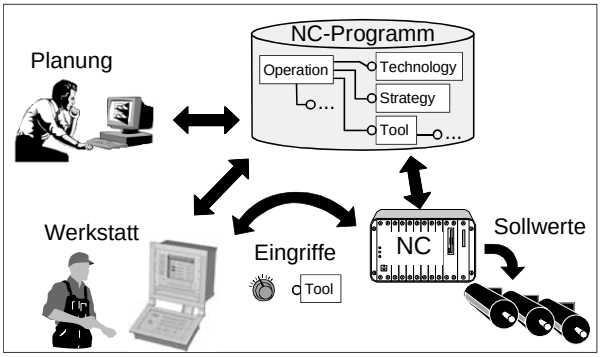
Darüber hinaus wird der Werker durch die NC-Steuerung sowie die Kapselung der Werkzeugmaschine vom mechanischen Bearbeitungsprozess entkoppelt [DAUD01]. Die Programmierung und die Verfolgung des Programmfortschritts erfolgen steuerungsorientiert über elementare Befehle, Positionsangaben und Technologieparameter. Die ablauf-, merkmals- und operationsorientierte Denkweise des Werkers wird durch die befehlsorientierte Programmierung nach DIN 66025 nicht, bzw. mit NC-Zyklen nur bedingt, unterstützt [WOLF01].

Heutige sequentielle Prozesskette



- Vielzahl spezialisierter Einzelsysteme
- Nur Teilinformationen an der Maschine verfügbar
- Optimierungen müssen manuell nachgeführt werden
- Eingeschränkte Eingriffsmöglichkeiten im Prozess

Konzept einer durchgängigen Anwenderunterstützung



+ Anwendergerechte Systeme für Planung und Werkstatt
+ Aufgabenbeschreibung durchgängig verfügbar
+ Direkter, durchgängiger Austausch von Änderungen
+ Umfassende Eingriffe in den Prozess möglich

Bild 7.1: Durchgängige, aufgaben- und prozessumfassende Anwenderunterstützung
Consistent, task and process oriented support of the operator

Diese Defizite elementarer, sequentieller Steuerungscodes (Bild 7.1 links) und der begrenzten Eingriffsmöglichkeiten löst das Gesamtkonzept ausgehend von der zuvor entwickelten NC-Programmierschnittstelle und dem Zugriff auf die steuerungsinterne NC-Programmumsetzung. Dabei muss der Anwender nicht mehr zwischen verschiedenen Systemen und Datenformaten wechseln oder die Bearbeitung unterbrechen. Stattdessen greift er von der Planung über das Einfahren, die Abarbeitungsverfolgung bis hin zur Prozessregelung durchgängig auf das die Bearbeitungsaufgabe umfassend beschreibende NC-Programm zu (Bild 7.1 rechts).

Die zur NC-Programmierung erforderlichen Funktionen, die Prozessvisualisierung sowie die werkerorientierten Interaktionsmechanismen zum bearbeitungsparallelen Prozesseingriff werden im Benutzungssystem der NC-Steuerung realisiert. Dieses besteht aus einem mit dem

Kern der NC-Steuerung gekoppelten Grundsystem (Kapitel 7.1), das dem Entwickler Basisfunktionen anbietet und in das er seine spezialisierten Anzeigen, Dialoge und Funktionen integrieren kann (Kapitel 7.2).

7.1. Grundsystem für einen modularen Aufbau

Core system for a modular software design

Analog zu dem beschriebenen Steuerungsaufbau wird auch das Benutzungssystem ausgehend von der aufgaben-, geometrie- und operationsorientierten NC-Programmierschnittstelle entwickelt. Anforderungskonform ist das System offen erweiterbar, um zu jedem NC-Programmelement standardisierte oder spezialisierte Oberflächen in die Funktionsbereiche der Mensch-Maschine-Schnittstelle (HMI, Human Machine Interface) zu integrieren. Schnittstellen zum NC-Kern stellen sicher, dass die Benutzerfunktionen mit den dort implementierten Fortschrittsmeldungen und Eingriffsfunktionen interagieren. In Bild 7.2 ist ein möglicher Aufbau eines solchen Benutzungssystems dargestellt.

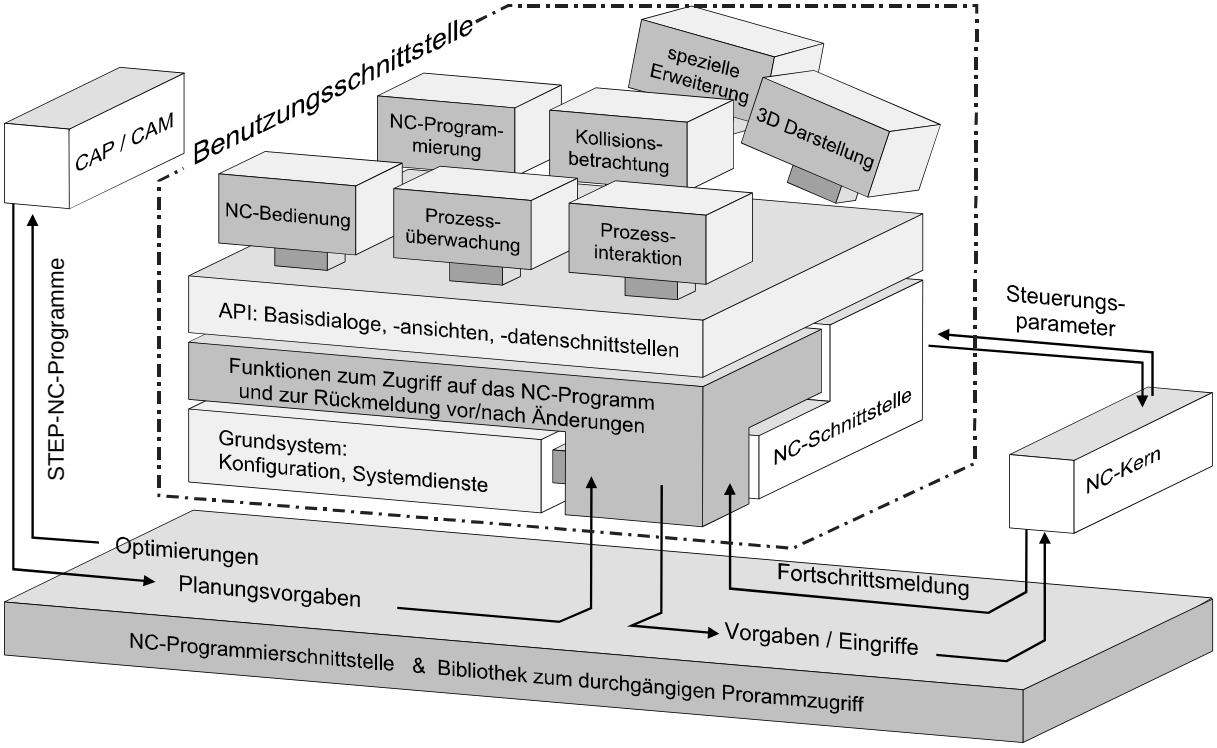


Bild 7.2: Aufbau und Schnittstellen des Programmier- und Benutzungssystems an der NC
Architecture and interfaces of the developed HMI

Auch wenn dieser Ansatz im Fokus der folgenden Detaillierung steht, so ist das Gesamtsystem keineswegs auf diesen Aufbau festgelegt. Prinzipiell können mehrere Benutzerfunktionen als eigenständige Applikationen entwickelt und zeitgleich eingesetzt werden. Intern werden sich jedoch die nachfolgend beschriebenen Teilfunktionen wiederfinden, da diese notwendig sind, um innerhalb des Gesamtsystems Programminformationen aufgaben- und prozessorientiert zu erstellen, auszutauschen und steuerungsseitig in die Bearbeitung einzugreifen. Der wesentliche Vorteil einer zentralen Plattform leitet sich dabei aus der Koordination möglicherweise gegenläufiger Programm- und Steuerungszugriffe ab. Zu Konzepten, wie solche Konflikte durch eine steuerungsseitige Integrationsplattform gelöst werden können, sei an dieser Stelle auf die Dissertation von *Kahmen* verwiesen [KAHM05].

7.1.1. Hardware und Betriebssystem

Hardware and operating system

Das HMI ist in der Regel auf einem eigenständigen, vom Steuerungskern getrennten, PC-basierten Rechner installiert. Ausgehend von seinem standardisierten Betriebssystem, Eingabehilfen, wie Tastatur, Maus und Softkeys und den Schnittstellen zum NC-Kern bietet sich der HMI-Rechner zur Installation eigener Applikationen an. Hierzu gehören NC-Programmiersysteme, Simulationen, Datenbanken zur Werkzeugverwaltung oder die Parameter- und Prozessvisualisierung.

Konzepte für eine hardware- und betriebssystemunabhängige Benutzeroberfläche, in die modular Funktionen integriert werden können, wurden unter anderem von *Wenk* vorgestellt. Er hat eine Java-basierte Oberfläche entwickelt, die mithilfe einer Virtual-Machine auf verschiedenen Hardware- und Betriebssystemen installiert und in einem Webbrowser aufgerufen werden kann [WENK06]. Sein Konzept geht von der konventionellen Anforderung aus, dass über die Oberfläche keine zeitkritischen Ein- und Ausgaben realisiert werden müssen. Folglich müssen sich das Java-HMI und seine Verbindung zum Steuerungskern nicht deterministisch verhalten. Als Grundsystem zur Integration von Überwachungs- und Regelsystemen mit hartem Echtzeitverhalten ist dieser Ansatz daher nicht geeignet.

Wie bereits dargestellt, sind die Benutzeroberflächen mehrheitlich auf Industrie PCs mit Windows-Betriebssystemen installiert. In Ausnahmen wird beispielsweise Linux als offenes Betriebssystem eingesetzt [WEIS07]. Auf dieser Grundlage ist eine systemangepasste und ressourcenoptimierte Programmierung in C++ oder C# nahe liegend. Für Microsoft Windows als auch für Linux werden leistungsfähige Entwicklungsumgebungen und Softwarebibliotheken angeboten, die den Entwickler bei der grafischen Umsetzung und beim Zugriff auf System- und Netzwerkfunktionen unterstützen. Das preemptive Multitasking der Betriebssysteme stellt sicher, dass einzelne Threads, d.h. Rechenprozesse, zyklisch im Prozessor abgearbeitet werden und ihn nicht blockieren. Durch software- oder hardwaretechnische Erweiterungen ist zudem ein hartes Echtzeitverhalten möglich, das den Threads neben definierten Prozessorzeiten auch deterministische Interrupts bereitstellt [EMDE07]. In Kombination mit einer Feldbusschnittstelle zur NC-Steuerung sind die systemtechnischen Anforderungen an das HMI-Konzept erfüllt. Der Werker kann den Prozess über die Benutzungsoberfläche mit einer determinierten Latenzzeit verfolgen und regeln. Indem eine echtzeitfähige Ethernet-Erweiterung eingesetzt wird (vgl. [DOYL04, SAUE04]) ist das Gesamtsystem zudem unabhängig von proprietären Bussystemen. Die Bandbreite des Ethernets ermöglicht den Austausch der umfangreichen, aus der entwickelten NC-Programmierschnittstelle abgeleiteten Datenstrukturen. Die Systemmodule lassen sich durchgängig softwarebasiert aufbauen.

Für den Einsatz der Microsoft Windows-Betriebssysteme sprechen ihr hoher Bekanntheitsgrad bei den Anwendern und die Möglichkeit, gängige Softwareapplikationen, wie beispielsweise zur Textverarbeitung, Tabellenkalkulation, CAD-Viewer oder Datenbanken, zu installieren. Die heutigen Arbeitsspeicher, Festplatten, Netzwerkanschlüsse und Dateiverwaltungssysteme erlauben den Austausch und das Öffnen großer Datensätze. Folglich kann vorausgesetzt werden, dass sich umfangreiche CAD-Datensätze und objektorientierte NC-Programme gemäß der im Rahmen dieser Arbeit mit erarbeiteten ISO 14649 Programmierschnittstelle über Anwendungen unter Windows-Betriebssystemen und PC-Hardware verarbeiten lassen.

7.1.2. Anwendergerechte, kontextorientierte Interaktion

User adequate, context oriented interaction

Die Vielzahl der prozessbestimmenden und im NC-Programm kodierten Parameter sowie die Offenheit, diese anwenderspezifisch zu erweitern und zu interpretieren, setzen umfangreiche Darstellungs- und Interaktionsmechanismen im Benutzungssystem voraus, die zudem kon-

textabhängig gesperrt, aktiviert und mit Parametern belegt werden müssen. Diesen Anforderungen werden die Steuertafel und die interaktive Anzeige heutiger HMI nicht gerecht. Ihre Funktionsweise wird nachfolgend kurz beschrieben, um im Anschluss ein Konzept zu entwerfen, mit dem sie für das Gesamtsystem nutzbar gemacht werden können.

Die **Maschinensteuertafel** dient dem Werker, um über Taster, Schalter und Drehregler direkt auf die Funktionen der Maschine und der Steuerung zugreifen zu können. Hierzu zählen beispielsweise die Betriebsart, Start und Stopp der Programmabarbeitung oder die Overrides zur Geschwindigkeitsmanipulation. In Abhängigkeit von der geforderten Reaktionsgeschwindigkeit und den Sicherheitsauflagen sind die Schalter hardwaretechnisch oder über echtzeitfähige Bussysteme mit der SPS und dem NC-Kern verbunden. Die Belegung der Schalter wird durch die Steuerungs- und Maschinenhersteller fest vorgegeben und lässt sich in der Regel nur zur Anwahl einzelner Verfahrachsen umschalten. Komplexe Abläufe, wie das Einwechseln eines alternativen Werkzeugs im laufenden NC-Programm, das Anfahren an einen definierten Punkt oder die Manipulation der Schnitttiefe sind heute über die Steuertafel nicht möglich. Dieser Umstand resultiert aus der fehlenden Kopplung zwischen der Steuertafel und den Vorgaben des NC-Programms sowie der elementaren Schalt- und Wegfunktionen des NC-Programms.

Deutlich flexibler ist die **interaktive Anzeige** der NC-Steuerung, das MMC (Man Machine Control). Das MMC dient dem Werker als zentrale Anzeige- und Eingabeeinheit. In ihr sind alle nicht echtzeitkritischen Funktionen, wie die Visualisierung der Prozess- und Stammdaten, die NC-Programmierung, die Ablaufsimulation und die Konfiguration der Maschinenparameter integriert. Die Teilfunktionen des MMC arbeiten als eigenständige Softwareapplikationen, die über ein serielles Bussystem auf die Daten und Funktionen der NC-Steuerung und der SPS zugreifen können. In der Regel basiert die MMC auf PC-Hardware und Windows-Betriebssystemen. Neben kostengünstigen, integrierten Lösungen, in denen das MMC und der NC-Kern auf einem Prozessor laufen, werden in leistungsfähigen Steuerungen gezielt getrennte Prozessoren eingesetzt, um den echtzeitkritischen NC-Kern nicht durch die teilweise ressourcenintensiven Applikationen des Bedienbereichs zu belasten [WECK01d].

7.1.2.1. Variable Funktionsbelegung der Hard- und Softwareschalter

Addressing functions flexibly to hard keys and soft keys

Zur Abbildung der erweiterten, aufgaben- und prozessorientierten Eingriffsoptionen in das Benutzungssystem bieten sich sowohl die Schalter und Regler der Maschinensteuertafel als auch die grafisch interaktiven Eingabemechanismen des MMC an. Die Schalter sind direkt greifbar und können blind benutzt werden, während der Werker den Bearbeitungsprozess weiter beobachtet. Das ist bei den anzeigeorientierten Interaktionsmechanismen des MMC nicht möglich. Bei ihnen erfolgt die Eingabe über Steuerelemente am Monitor, sogenannte Softcontrols, die situationsabhängig angezeigt werden und mit grafischen Darstellungen das Prozessverständnis des Werkers unterstützen. Typische Controls sind Textfelder oder animierte Schalter, Schieber, Check-Boxen und Auswahlmenüs. Die Eingabe erfolgt entweder über die Softkeys am Rand der Anzeige, eine Maus oder durch Berührung an einem Touchscreen. Nachteilig gegenüber den Schaltern der Steuertafel ist, dass der Werker diese Elemente nicht greifen kann, sondern sie zur Bedienung sehen muss. Um beide Mechanismen parallel nutzen zu können, wird für das HMI-Konzept eine flexible, situationsabhängige Zuordnung der Eingriffsfunktionen entwickelt. Die hardwaretechnische Trennung wird dabei durch das Grundsystem des HMI überbrückt. Der Werker kann so die einzelnen Parameter wahlweise grafisch unterstützt am Bedienfeld oder mit Blick auf den Prozess an der Steuertafel manipulieren.

In Bild 7.3 ist das Ziel des Konzepts dargestellt. Hierbei wird die Anzeige des HMI genutzt, um dem Werker in Abhängigkeit von der Bearbeitungsaufgabe Alternativwerkzeuge vorzuschlagen oder zum Ende des Arbeitsschritts einen definierten Halt anzubieten. Ferner kann er

die Schnittgeschwindigkeit, Schnitttiefe und Werkzeugüberdeckung variieren. Die Overrides der Steuertafel sind mit den im MMC angezeigten Werten gekoppelt. Manipuliert der Werker beispielsweise die Spindeldrehzahl S, wird automatisch die Schnittgeschwindigkeit v_c in der Anzeige aktualisiert. In umgekehrter Richtung wird eine Aktualisierung möglich, indem anstelle fester Skalen Dioden oder Displays in die Tafel integriert werden.

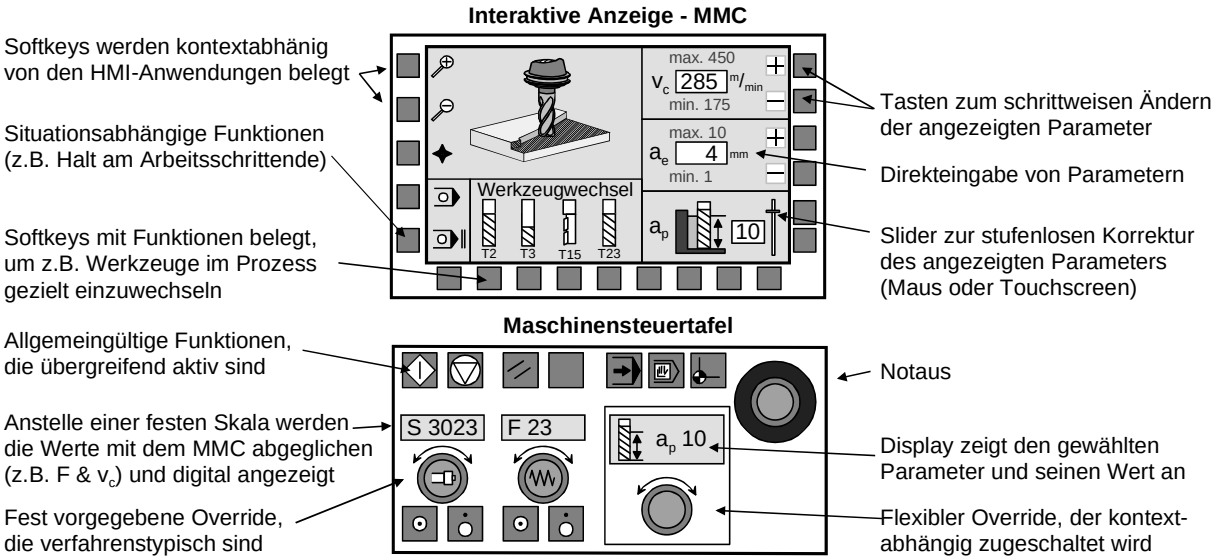

Interaktive Anzeige - MMC
Softkeys werden kontextabhänig von den HMI-Anwendungen belegt
Situationsabhängige Funktionen (z.B. Halt am Arbeitsschrittende)
Softkeys mit Funktionen belegt, um z.B. Werkzeuge im Prozess gezielt einzuwechseln
max. 450
v_c 285 m/min
min. 175
max. 10
a_e 4 mm
min. 1
Werkzeugwechsel
T2 T3 T15 T23
a_p 10
Tasten zum schrittweisen Ändern der angezeigten Parameter
Direkteingabe von Parametern
Slider zur stufenlosen Korrektur des angezeigten Parameters (Maus oder Touchscreen)
Maschinensteuertafel
Allgemeingültige Funktionen, die übergreifend aktiv sind
Anstelle einer festen Skala werden die Werte mit dem MMC abgeglichen (z.B. F & v_c) und digital angezeigt
Fest vorgegebene Override, die verfahrenstypisch sind
S 3023
F 23
a_p 10
Notaus
Display zeigt den gewählten Parameter und seinen Wert an
Flexibler Override, der kontextabhängig zugeschaltet wird

Bild 7.3: Hard- und Softwareschalter zur Benutzung der Steuerungsfunktionen
Hard keys and soft keys to operate the numerical control

Systemtechnisch setzt diese Lösung voraus, dass die Schalter der Steuertafel nicht starr mit den Funktionen des HMI und denen des NC-Kern gekoppelt sind. Anderenfalls wäre eine große, nicht sinnvoll anzuordnende und für den Werker unübersichtliche Anzahl von Schaltern zur Realisierung notwendig. Diese Anzahl leitet sich aus den zu manipulierenden Parametern der offen erweiterbaren NC-Programmierschnittstelle und den darauf steuerungsseitig aufbauenden NC-Workingsteps mit ihren spezifischen Eingriffsfunktionen ab. Das konzipierte Benutzungssystem sieht daher eine variable, softwarebasierte Zuordnung zwischen den verschiedenen Stellgrößen und den Schaltern der Steuertafel sowie den Softkeys des MMI vor.

Die Verwaltung der Zuordnung übernimmt das Grundsystem des HMI und steuert diese kontextabhängig. Hierzu bietet sich eine generische, variablenbasierte Schnittstelle an. In dieser wird jeder Schalter, Override und Softkey auf eine Variablenstruktur abgebildet. Die einzelnen Variablen beschreiben den Typ des Schalters und seinen aktuellen Wert. Im Fall der Override werden zusätzlich die Inkrementgröße und die zulässigen Grenzwerte definiert. Mithilfe einer Konfigurationsdatei werden die Variablen mit den Funktionen der HMI-Anwendungen und den Schnittstellen des NC-Kerns verknüpft. Denkbare Konfigurationskriterien sind die Betriebsart, die im NC-Programm kodierten Operations- und Technologiedatenelemente oder die steuerungsseitigen NC-Workingsteps. Ändert sich beispielsweise während der Bearbeitung die Operation, so passt das HMI automatisch die Funktionen der verschiedenen Schalter und Override an. Dieser Ansatz stellt dem Werker jederzeit die steuerungsseitig verfügbaren und für den Prozess geeigneten Eingriffsfunktionen bereit.

Neben expliziten Parametern lassen sich den Schaltern und Softkeys auch komplexe Funktionsabläufe zuweisen. In Bild 7.3 werden beispielsweise die horizontalen Softkeys mit Werkzeugen vorbelegt. Betätigt der Werker einen der Schalter, so löst er eine Routine aus, um die Bearbeitung kontrolliert zu unterbrechen, das aktuelle Werkzeug freizufahren und auszuwechseln sowie das neue Werkzeug automatisch an die adaptierten Bahnen zuzustellen.

7.1.2.2. Ausnahme bei sicherheitskritischen und übergreifend gültigen Funktionen

Restrictions for safety relevant functions or general functions

Dennoch sollten einzelne Schalter und Override von dem Konzept einer flexiblen, operationsgesteuerten Belegung ausgeschlossen werden. Hierzu gehören der Notaus oder Funktionen, wie die Betriebsartenwahl und Maschinengrundfunktionen. Diese Einschränkung stellt sicher, dass kritische und allgemeingültige Funktionen durchgängig verfügbar sind und sie nicht aufgrund von Softwarefehlern deaktiviert werden. Ferner unterstützt sie den Werker, der diese spezifischen Schalter und Override immer am gleichen Ort wiederfindet. Er kann sie somit intuitiv greifen und schneller reagieren.

7.1.3. Modular erweiterbare HMI-Funktionen

Modular extendable HMI functionalities

Die Anforderung, den Werker bei der Arbeit mit der NC-Steuerung kontextabhängig und intuitiv zu unterstützen, übernehmen in Abhängigkeit von der NC-Funktionalität und den Anwenderanforderungen spezialisierte Applikationen. Diese werden über die offenen HMI-Schnittstellen der Steuerungshersteller in die Oberfläche integriert und mit den Datenbanken, Funktionen und Parametern der NC-Steuerung gekoppelt. Der Aufruf der verschiedenen Applikationen erfolgt über die Softkeys der Benutzungsoberfläche oder automatisiert in Abhängigkeit von dem Betriebszustand der NC-Steuerung. Ihr Funktionsumfang wird durch den Steuerungs- und Systementwickler vorgegeben. Er reicht von der Prozessanzeige, über NC-Programmiersysteme bis hin zu interaktiven Oberflächen zur Prozessverfolgung und -regelung. Einen standardisierten HMI-Aufbau gibt es heute nicht, gleichwohl im Projekt HÜMNOS ein Styleguide entwickelt wurde, der eine Referenzarchitektur für die einzelnen Funktionseinheiten des HMI definiert [HUEM98]. Initiativen, wie das Global HMI haben sich auf die Anbindung der Benutzungsoberfläche an die NC-Steuerung beschränkt [MICH00]. Damit fehlen offene Schnittstellen, über die die einzelnen HMI-Applikationen untereinander Daten austauschen und Funktionen aufrufen können. Beispielsweise kann heute ein NC-Programmiersystem, wie ShopMill der Fa. Siemens, nicht um eigene NC-Programmelemente, Anzeigen, Dialoge und Interaktionsmechanismen zu deren Programmierung oder zur Prozessregelung erweitert werden. Ein modularer Systemaufbau, der die Interaktion der verschiedenen HMI-Applikationen sicherstellt und zudem eine offene, erweiterbare NC-Programmierschnittstelle durchgängig von der Datenübernahme, über die Anzeige, die Programmierung bis hin zur Prozessverfolgung und Regelung unterstützt, ist Ziel dieser Arbeit und wird als Basis für das HMI-Konzept entwickelt.

7.1.3.1. Schnittstellen zur modularen Systemerweiterung

Interfaces for modular system enhancements

Analog zu den konventionellen HMI muss das Benutzungskonzept ebenfalls die Möglichkeit anbieten, steuerungs- und anwenderspezifische Funktionen in die Oberfläche zu integrieren und diese mit den verfügbaren Eingabemechanismen und Funktionen der NC-Steuerung zu verbinden. Wie in Bild 7.2 dargestellt ist, übernimmt diese Aufgabe das Grundssystem des HMI. Darüber hinaus dient das Grundsystem als Plattform, über die die HMI-Funktionsmodule untereinander Daten austauschen und ihre Funktionen gegenseitig aufrufen können.

Theoretische wäre eine Kopplung der Funktionsmodule über Mechanismen wie CORBA möglich und damit unabhängig von dem eingesetzten Betriebssystem und der Programmiersprache. Da jedoch die Mehrzahl der HMI auf einem PC installiert ist und unter Windows Betriebssystemen läuft, ist es nahe liegend die Windows-Mechanismen zu nutzen:

- Anwendungen (*.exe)

- Bibliotheken (*.dll)
- ActiveX Controls (*.ocx)

Anwendungen

Konventionell wird der Funktionsumfang eines HMI über eigenständige Anwendungen erweitert. Diese werden aus der Benutzungsoberfläche gestartet und ihre Fenster in der Steuerungsoberfläche zur Anzeige gebracht. Jede der Anwendung wird als eigenständiger Prozess mit einem in sich abgeschlossenen Speicherbereich gestartet. Zum Datenaustausch mit einer anderen Applikation wird das Betriebssystem, die Festplatte oder ein gemeinsamer Speicherbereich genutzt, den eine eingebundene Bibliothek bereitstellt. Ein Beispiel für die letzte Variante, das sogenannte Shared Memory, ist die OSACA Kommunikationsplattform. Ihre Bibliothek stellt neben dem Speicherbereich auch Methoden zum kollisionsfreien Zugriff auf Daten, Prozesse und Events zur Benachrichtigung bei Änderungen bereit.

Der Sinn einer Anwendung ist jedoch, dass sie eine in sich geschlossene Funktionseinheit umfasst, die eigenständig gestartet und beendet werden kann. Daraus resultieren Probleme, wenn beispielsweise die Anwendung geschlossen wird, während noch externe Zugriffe auf die Daten und Funktionen erfolgen. Es ist daher sinnvoller, die erweiternden Funktionsmodule des HMI nicht in Form einer Vielzahl miteinander zu koppelnder Anwendungen zu implementieren, sondern als Bibliotheken, die von einer zentralen Anwendung, dem Grundsystem des HMI, geladen, verwaltet und angesteuert werden.

Bibliotheken

Die Bibliotheken können analog zu den Anwendungen komplexe Funktionsumfänge, grafische Anzeigen und Klassenstrukturen enthalten. Sie sind jedoch zur Ausführung auf eine Anwendung angewiesen, die sie lädt und die ihre internen Funktionen über das Applikationsinterface (API) der Bibliothek aufruft. In der API muss nicht die gesamte Implementierung offengelegt werden. In der Regel enthält sie nur die auszutauschenden Datenkonstrukte und die Funktionsaufrufe. Das spezifische Wissen des Entwicklers wird hinter der API geschützt.

Es lassen sich zwei Arten von Bibliotheken unterscheiden: statische und dynamische Bibliotheken. Die NC-Programmierschnittstelle wird beispielsweise in Form einer statischen Bibliothek in das Grundsystem des HMI eingebunden. Sie wird mit der Applikation geladen und steht damit durchgängig zur Verfügung. Die Funktionserweiterungen werden dagegen je nach Bedarf dynamisch geladen und entladen. Dadurch belasten zum einen inaktive Erweiterungen nicht den Arbeitsspeicher und zum anderen wird die Anforderung erfüllt, dass der Anwender die Funktionsmodule des HMI zur Laufzeit hinzufügen, austauschen und konfigurieren kann, ohne die Anwendung neu starten zu müssen.

Ein weiterer Vorteil, die HMI-Erweiterungen in Form von Bibliotheken zu kapseln, resultiert daraus, dass eine Bibliothek nicht in einem eigenen Prozess läuft. Bei einem preemptiven Multitasking, wie es beispielsweise unter Windows NT oder Windows XP eingesetzt wird, wird die verfügbare Prozessorzeit in Zeitscheiben unterteilt und den Prozessen sequentiell oder nach Priorität über einen Scheduler zugewiesen. Die Ausführung der Bibliotheksfunktionen wird dagegen von der aufrufenden Anwendung kontrolliert. Sie kann die Funktionen gezielt starten und ihnen einen parallel laufenden Thread zuweisen, den sie über Semaphoren und Events jederzeit unterbrechen oder aufrufen kann. Während dieser Mechanismus für konventionelle HMI mit ihren nicht-echtzeitkritischen Anzeigemechanismen eine untergeordnete Bedeutung hat, ist er für die entwickelte Prozessregelung essentiell, um schnell reagieren zu können.

ActiveX Controls

Ein ActiveX ist eine Softwarekomponente, die auf dem Microsoft COM Standard aufsetzt und analog zu einer Bibliothek nur innerhalb einer anderen Anwendung ausgeführt werden

kann, indem sie auf deren Funktionen und Daten zugreift. In der Steuerungstechnik bieten unter anderem Bosch Rexroth und Siemens Schnittstellen an, um ActiveX Controls in ihre Oberflächen zu integrieren und so kundenspezifische Parameteranzeigen zu implementieren [BOSC04]. Im Gegensatz zu einer Bibliothek stellt ein ActiveX Control jedoch keine Schnittstellen nach außen zur Verfügung. Damit ist die geforderte Interaktionsfähigkeit nur bedingt gegeben. Im Rahmen des Konzepts bieten sich ActiveX Controls daher nur für einfache Interaktionsaufgaben an, wie beispielsweise Schalter und Regler, die nicht von anderen Modulen aufgerufen werden müssen.

7.1.3.2. Basisfunktionen als Grundlage für Spezialisierungen

Core functions as a framework for specialized functions

Ausgehend von den Mechanismen zur Systemerweiterung kommen dem Grundsystem zwei zentrale Funktionen zu. Zum einen muss das Grundsystem die Erweiterungsmodule laden, konfigurieren, verwalten und ihr Laufzeitverhalten kontrollieren. Darüber hinaus dient das Grundsystem als Plattform, über die die HMI-Erweiterungen interagieren und koordiniert auf das NC-Programm sowie die Schnittstellen zur NC-Steuerung zugreifen.

Laden und steuern der Erweiterungsmodule

Die Anforderung, den Funktionsumfang des HMI flexibel erweitern und konfigurieren zu können, lässt sich nicht über statisch verknüpfte oder für den Anwender unzugänglich registrierte Bibliotheken realisieren. Eine offene Lösung sind Konfigurationsdateien, in denen die verfügbaren Schnittstellen des HMI-Grundsystems beschrieben sind und in denen die Bibliotheken mit ihren Aufrufparametern textbasiert zugewiesen werden können. Die Konfiguration kann entweder händisch in einem Editor oder mithilfe eines eigenen HMI-Funktionsmoduls grafisch interaktiv erfolgen. Während des Systemstarts oder initiiert durch den Anwender liest das Grundsystem die Konfigurationsdatei ein, entlädt alle nicht mehr relevanten Bibliotheken und ActiveX Controls bevor die neuen Elemente geladen, mit den Basisfunktionen verbunden und initialisiert werden. Die API der einzelnen Bibliotheken wird vom Grundsystem vorgegeben, damit die Schnittstellen eindeutig aufgerufen werden können. Zu der API gehören unter anderem Anzeigefenster, Menüleisten, Dialoge, Schnittstelleninterfaces zur NC-Steuerung, Berechnungs- und Verifikationsroutinen. Durch sie werden der grafische Aufbau und der Funktionsumfang des HMI definiert.

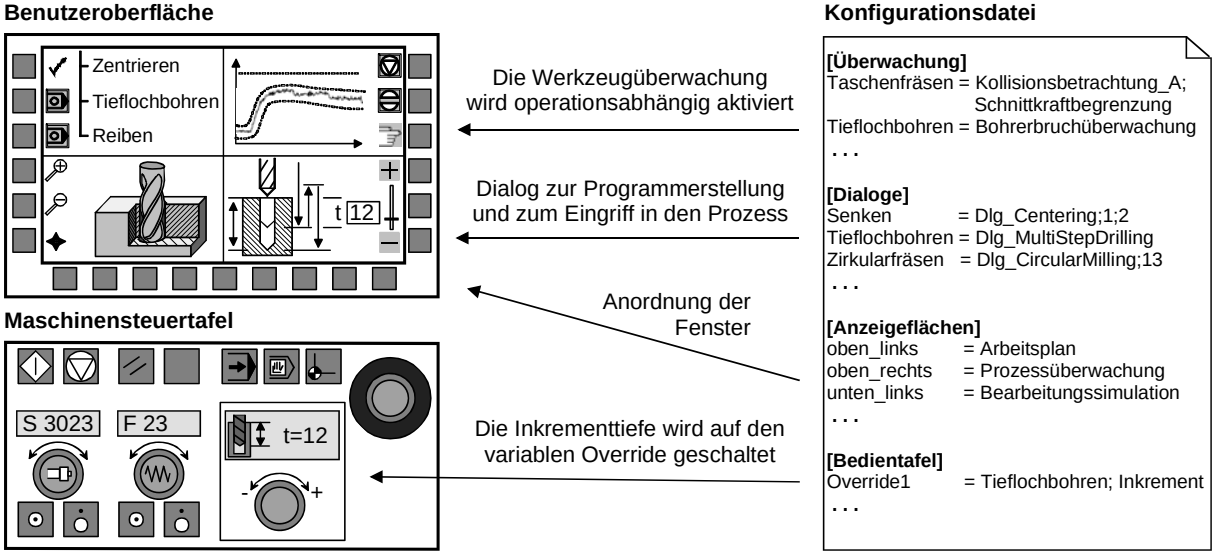


Bild 7.4: Konfiguration und Integration der Erweiterungsmodule in das Benutzungssystem

Configuration and integration of additional modules into the user interface

Hinter jeder der vorgegebenen Schnittstellen steht im Grundsystem eine Basisfunktion. Diese rufen die in den Konfigurationsdateien aufgelisteten Bibliotheken auf, laden die darin implementierten Schnittstellenfunktionen und speichern sie entsprechend ihrer Konfiguration als Funktionszeiger innerhalb des Grundsystems. Das HMI erlaubt je nach Schnittstelle das Speichern mehrerer Zeiger. Darüber ist es beispielsweise möglich, in den Anzeigeflächen des HMI wahlweise verschiedene Erweiterungsmodule zur Anzeige zu bringen, oder aus einer Vielzahl von Bibliotheken Dialoge zur NC-Programmierung zu laden, die dann in Abhängigkeit von dem angewählten NC-Element aufgerufen werden (Bild 7.4). Nicht konfigurierte Schnittstellen belegt das Grundsystem mit Standardfunktionen oder aber mit einer entsprechenden Warnmeldung. Zudem werden sie in der Konfigurationsdatei angezeigt.

Mechanismen zur Interaktion der Module

Die zweite zentrale Funktion des Grundsystems besteht in der Bereitstellung einer Funktionsschnittstelle, über die auf das NC-Programm, die NC-Steuerung und die anderen Module innerhalb des HMI zugegriffen werden kann. Um die Anforderung nach einer flexiblen Erweiterbarkeit der NC-Programmierschnittstelle und der Systemfunktionen zu gewährleisten, bietet das Grundsystem hierzu generische Funktionen an.

Im Einzelnen gehören zu diesen Funktionsmechanismen

- die NC-Programmierschnittstelle als zentrale Datenbasis zur systemweiten Interaktion
- ein Nachrichtensystem, über das beliebige Anweisungen und Daten zwischen den Anwendungen des HMI und mit den Steuerungsmodulen ausgetauscht werden,
- ein Schnittstellenserver zur Kopplung mit der NC-Steuerung und externen Systemen

Die einzelnen Mechanismen werden als Basisfunktionen in Form einer Schnittstellenbibliothek bereitgestellt. Der Systementwickler muss sich nicht um die spezifische Ausprägung der Schnittstellen kümmern. Er bindet die Basisschnittstellen in das von ihm entwickelte Modul ein, erhält damit Zugriff auf die Systemplattform und überlässt die Anpassung dem Grundsystem. Zur Laufzeit werden sie durch die spezialisierten Funktionen der in der Konfigurationsdatei eingetragenen Bibliotheken überladen.

Zum Beispiel wird die steuerungsspezifische Anbindung in Form einer Bibliothek gekapselt, deren Funktionen sich mit den Schnittstellen der NC-Steuerung verbinden. Im Fall der Siemens Sinumerik 840D würde dazu auf den NC-DDE Server zugegriffen. Zur Bosch MTX wird greift das Interface die NC-Schnittstelle (NCS) zu und zur Anbindung an eine OSACA Steuerung wird das Grundsystem der Benutzeroberfläche als sogenanntes Architekturobjekt mit der NC-Steuerung verbunden. Das Architekturobjekt verbindet sich mit der NC-Steuerung über Server- und Klientobjekte. Hierüber können Variablen ausgetauscht und Funktionen aufgerufen werden.

7.1.4. Das NC-Programm als einheitliche Datenbasis

The NC-program as a common data base

Das vorgestellte Konzept der NC-Programmierschnittstelle sieht vor, dass im NC-Programm alle Daten zur Prozessplanung, Bearbeitungsausführung und Optimierung zusammenhängend gespeichert und ausgetauscht werden können. Informationsverluste durch inkompatible Zusatzdaten oder Konvertierungen mit Postprozessoren werden auf der Basis des durchgängig einsetzbaren Datenmodells verhindert. Analog zu den Umsetzungsalgorithmen in der NC-Steuerung, die sich an den Elementen und dem Aufbau der Programmierschnittstelle orientieren, muss auch in der Benutzerschnittstelle diese Systematik und die darauf aufbauende Offenheit beibehalten werden. Die Funktionsstrukturen und der Datenfluss des HMI orientieren sich daher an dem Aufbau und dem Inhalt des NC-Programms. In seinen Datenelementen ist

die Bearbeitungsaufgabe schrittweise von der Planung bis zur optimierten Steuerungsanweisung detailliert.

7.1.4.1. Konsistente, durchgängig verfügbare NC-Programmverwaltung

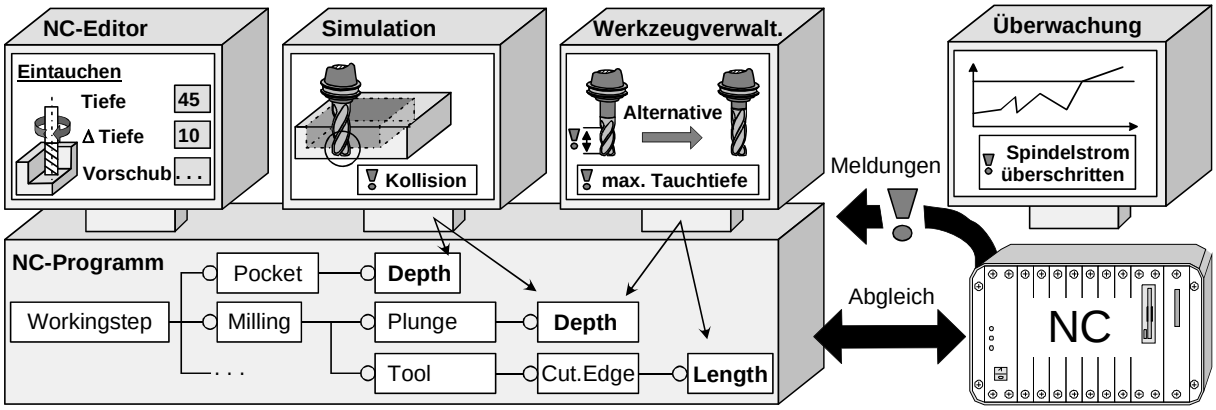
Consistent, universal NC-program management

In konventionellen HMI liest jede Applikation das NC-Programm individuell ein. Dies ist jedoch nur möglich, wenn das Programm in der NC-Steuerung abgewählt ist und keine anderen Anwendungen darauf zugreifen. Folglich ist ein paralleler Zugriff auf die Programmdaten nicht möglich. Die verschiedenen Anwendungen zur Programmerstellung, -optimierung und bearbeitungsparallelen Programmkorrektur können nicht zeitgleich interagieren. Änderungen werden nicht unmittelbar von allen in das HMI und den NC-Kern integrierten Anwendungen auf Fehler und ihre Umsetzbarkeit hin überprüft, in die laufende Bearbeitung übernommen und grafisch zur Anzeige gebracht. Zudem arbeiten viele der Programmiersysteme auf der Basis eigener Datenstrukturen und generieren erst über einen Postprozessor das NC-Programm. Dabei gehen übergeordnete Planungs- und Geometrieinformationen verloren. Diese Einschränkung löst das HMI-Konzept mithilfe einer zentralen Datenhaltung. Sie ist die Grundlage, um bearbeitungsparallel mit verschiedenen HMI Funktionen auf das NC-Programm zuzugreifen und sicherzustellen, dass im Gesamtsystem alle Programminformationen durchgängig und konsistent bereitgestellt werden.

Gemeinsamer Zugriff auf ein zentral verwaltetes NC-Programm

Der Kern dieses Ansatzes wird in Form einer zentral verfügbaren Funktionseinheit zum Einlesen, Verwalten und Schreiben des NC-Programms realisiert. Sie hat exklusiven Zugriff auf die physikalische Datei des NC-Programms und stellt über Softwareschnittstellen seinen Inhalt kontrolliert zur Verfügung. Alle Module, die den Inhalt des NC-Programms auswerten, darstellen und manipulieren wollen, müssen das über diese Funktionseinheit tun. Dieser Ansatz ist vergleichbar mit einer Datenbank, auf die verschiedene Clients parallel zugreifen können. Im Fall des Steuerungskonzepts gehören zu den Clients die NC-Steuerung, NC-Programmiermodule, Simulationen, Fortschrittsanzeigen und die Anwendungen zur Prozessregelung. Für STEP konforme Datenfiles sind kommerzielle Bibliotheken am Markt verfügbar. Exemplarisch sei hier auf die Softwarebibliotheken von STEP Tools verwiesen, einem Projektpartner mit dem die beschriebene Programmierschnittstelle entwickelt wurde [STEP06] und auf der die STEP-NC Planungssysteme der Universität Stuttgart zum Drehen aufsetzen [HEUS05]. Dieser Ansatz ist nicht nur auf lokale Systeme beschränkt. Eine Serverapplikation kann über Ethernet ganze Unternehmensbereiche von der Konstruktion über die Planung bis zur Fertigung bedienen [ALBE00]. Alle Programmänderungen und -ergänzungen erfolgen über die Funktionseinheit und müssen konform mit der darin implementierten NC-Programmierschnittstelle sein. Jeder Eingriff in die Programmvorgaben wird zentral registriert und an die verbundenen Anwendungen gemeldet. Deren interne Funktionen prüfen die Änderungen auf Fehler oder Grenzwertüberschreitung. Nur wenn keine Fehler an die zentrale Programmverwaltung zurückgemeldet werden, wird die Änderung systemweit umgesetzt.

Die in Kapitel 0 definierten Anforderungen einer durchgängigen Nutzung aller Systemfunktionen zur Unterstützung des Werkers und Absicherung einer fehlerfreien Bearbeitung werden erfüllt. Die durch das Unterbrechen, Schließen und erneute Laden des NC-Programms entstehenden Zeitverluste entfallen. Die parallel im HMI aktiven Anwendungen ergänzen sich durch ihre Funktionen, so dass beispielsweise der Effekt einer Programmänderung gleichzeitig in einer Simulation angezeigt werden kann und die Werkzeugverwaltung meldet, welche Werkzeuge für den neuen Bearbeitungsfall zur Verfügung stehen, Bild 7.5.



Die Teilsysteme greifen auf eine gemeinsame NC-Programmdatenbasis zu und interagieren über diese.
Die Funktionen ergänzen sich gegenseitig, so dass das Gesamtsystem den Anwender umfassend unterstützt.

Bild 7.5: Gleichzeitiger Programmzugriff als Basis sich ergänzender HMI-Funktionen
Simultaneous access to one NC-program by different HMI-modules

Beim gleichzeitigen Schreiben und Lesen mehrerer Anwendungen auf ein zentral verwaltetes NC-Programm treten Zugriffskonflikte auf. Dieses Problem wird durch Schnittstellenfunktionen umgangen, die sicherstellen, dass nur vollständige Daten ausgelesen werden, sowie ein Schreiben erst nach Abschluss aller laufenden Leseoperationen ausgeführt wird. Softwaretechnisch wird diese Anforderung durch eine objektorientierte Kapselung der NC-Programmelemente mit objektinternen Zustandsvariablen und Wartefunktionen realisiert. Mittels der Zustände wird zudem der Programminhalt gesperrt, um beispielsweise die Programmausführung vor einem erkannten Fehler zu stoppen oder aber Eingriffe nur in Abhängigkeit von dem Umsetzungs- und Abarbeitungsfortschritt der Steuerung zuzulassen.

Der derart in Objekten gekapselte Programminhalt wird folglich nicht direkt von einer einzelnen Applikation geändert. Er fragt eine Änderung an, die dann erst durch alle verbundenen Systeme und deren Funktionen geprüft wird. Während dessen wechselt der Objektzustand auf „in Änderung“ und wechselt erst nach positiver Rückmeldung in den Grundzustand, in dem er die Änderung übernimmt. Unstimmigkeiten werden mit der zurückgemeldeten Information dem Werker zur Anzeige gebracht oder an ein übergeordnetes Funktionsmodul zur Entscheidung übertragen.

Durchgängige Datenbasis bei verteilten Systemen

Bei hardwaretechnisch getrennten Systemen ist der zuvor beschriebene Ansatz eines zentralen, das HMI und den NC-Kern übergreifenden Programmspeichers (vgl. Bild 7.2) nicht umsetzbar. Auch klassische Konzepte mit über Ethernet verbundenen Datenservern bieten sich nicht an, da sie die Datenelemente nicht deterministisch bereitstellen können und daher eine unterbrechungsfreie NC-Programmverarbeitung nicht garantiert werden kann.

Ein Lösungsansatz besteht darin, das NC-Programm in mehrere, systemspezifische Datenspeicher zu laden. Untereinander werden diese Speicher über den Austausch der geänderten oder ergänzten Programmelemente abgeglichen. Ausgehend von einer objektorientierten Umsetzung der NC-Programmierschnittstelle lassen sich die einzelnen Programmelemente als Datenobjekte kapseln. Innerhalb der Datenspeicher wird ihr Bezug zu dem hierarchisch aufgebauten NC-Programm über die eindeutige Referenz jedes Objekts rekonstruiert. Eine solche Referenz wäre im Fall der Programmcodierung nach ISO 10303-21 die Identifikationsnummer eines STEP-Datenobjekts, z.B. „#405“.

Diese Funktionen lassen sich in Kombination mit dem in Kapitel 5.1 vorgestellten Konzept einer generischen, als Bibliothek gekapselten NC-Programmierschnittstelle zentral in den

Grundsystemen der Benutzungsoberfläche und des NC-Kern implementieren. Nach jeder Erweiterung der Programmierschnittstelle wird diese Bibliothek neu erstellt und von den einzelnen Anwendungen eingebunden. Dabei verhält sie sich neutral gegenüber den individuellen Bedeutungen der Programmelemente. Ihre Logik beschränkt sich auf die Unterscheidung der Basistypen, wie Text, Zahlenwerten oder Vektoren und Matrizen, die zu den verschiedenen Parametern definierten Regeln (z.B. dass ein Richtungsvektor kein Null-Vektor sein kann) und die Funktionen zur Zugriffskontrolle. Ferner bietet sich ausgehend von einer Normierung, z.B. der ISO 14649, die Unterscheidung allgemeingültiger Programmelemente an. Sie ermöglichen einen schnelleren Zugriff auf vordefinierte Strukturen, wie einen Arbeitsplan, die Werkstückgeometrie oder die Werkzeugliste.

7.1.4.2. Aktualisierung und Funktionsaufruf über NC-Programmelemente

Feedback and function calls originated by the processed NC-program elements

Die zentrale Verwaltung des NC-Programminhalts und ihre Synchronisation mit der Programmverarbeitung des NC-Kerns liefert die Grundlage für eine anwendungsübergreifend konsistente, jederzeit verfügbare Datenbasis. Diese reicht jedoch nicht aus, um aktiv auf Änderungen zu reagieren oder aber zwischen den Funktionen des Gesamtsystems aufgaben-, merkmals- und prozessorientiert zu interagieren. Die auf der NC-Programmierschnittstelle aufgebaute Datenverwaltung muss um zusätzliche Mechanismen zur Änderungsverfolgung und zum Funktionsaufruf ergänzt werden.

Die konsequente Kapselung der einzelnen Programmvorgaben und -elemente in Softwareobjekten mit definierten Zugriffsfunktionen ermöglicht das Erkennen von Änderungen und die Benachrichtigung der mit der Programmverwaltung verbundenen Anwendungen. Dazu ist das Lesen und Schreiben des Programminhalts nur über die Methoden der einzelnen Objekte möglich. Die softwaretechnische Umsetzung der Lesemethoden bietet die Möglichkeit, einen Verweis auf die lesende Anwendung zu speichern. Auf diese Weise merkt sich das Objekt, von wem seine Daten aktuell benutzt werden. Erst durch den Aufruf von Abmeldemethoden werden die Merker wieder freigegeben. Dieses Prinzip ist in der Softwareentwicklung gängig, um mittels intelligenter Zeiger, den Smart Pointern, zu verhindern, dass Speicherbereiche gelöscht werden, während externe Funktionen noch auf sie zugreifen. Erweitert man diesen Ansatz um das Speichern sogenannter Rückruffunktionen, so lassen sich aus dem Objekt heraus Funktionen innerhalb der zugreifenden Anwendungen aufrufen. Eine gezielte Klassifizierung der Rückruffunktionen erlaubt es, später gezielt vor oder nach einer Änderung am Objektinhalt mit den verschiedenen Anwendungen zu interagieren. Die Koordination der Funktionsaufrufe ist die Aufgabe der Schreib- oder Löschmethoden.

Soll eine Programmvorgabe geändert werden, so wird die Schreibmethode des dazugehörigen Objekts aufgerufen. Diese Methoden überprüfen zunächst intern, ob die Änderung syntaktisch richtig ist und die für den Wert gültigen Grenzwerte eingehalten wurden. Diese Grundfunktion hängt im Wesentlichen von dem Datentypen des NC-Programmelements ab und nicht von seiner auf die Bearbeitungsaufgabe bezogenen Bedeutung. Folglich kann sie ausgehend von der NC-Programmierschnittstelle generisch als Methode der Objekte implementiert werden. Liefert die interne Überprüfung keine Fehler, so werden die gespeicherten, vor der Umsetzung einer Änderung aufzurufenden Rückruffunktionen abgearbeitet. Beispiele hierfür wären Kollisionskontrollen, Prozesssimulationen, der Abgleich mit der steuerungsintegrierten Werkzeugdatenbank und den Maschinenparametern sowie die Überprüfung, ob der aktuelle Bearbeitungsfortschritt eine Änderung ggf. nicht mehr zulässt. Erst wenn diese Funktionen keine Fehler oder Einschränkungen zurückmelden, wird die Änderung aktiv. In einem letzten Schritt wird der neue Objektinhalt über die noch ausstehenden Rückruffunktionen an die verschiedenen Anwendungen gemeldet. Die Programmverarbeitung, Prozessüberwachung und die Bearbeitungssimulation des HMI werden aktualisiert.

Diese Lösung stellt sicher, dass zum einen nur gültige Änderungen durchgeführt werden und zum anderen die Umsetzung im HMI sowie in der NC-Steuerung jederzeit konsistent ist.

7.1.5. Nachrichtensystem zum Datenaustausch und Funktionsaufruf

Messaging system to exchange data and function calls

Da die Funktionen des HMI nicht durch einzelne, in sich abgeschlossene Applikationen realisiert werden, müssen sie von einer Basisapplikation aufgerufen und gesteuert werden. Diese Aufgabe übernimmt das Grundsystem. Einen vergleichbaren Systemaufbau findet man heute bei fast allen CAD/CAM Systemen. Aus der Sicht des Anwenders startet er das CAD/CAM-System als eine in sich geschlossene Anwendung. Die darin verfügbaren Teilfunktionen werden jedoch aus Bibliotheken, die je nach Ausbaustufe installiert werden, hinzugeladen. Den Aufruf und die Interaktion der Bibliotheken stellen Funktionsschnittstellen und Nachrichtensysteme sicher.

Für das Benutzungssystem einer NC-Steuerung sind nur wenige Funktionselemente standardisiert. Die Spezifikationen aus den Initiativen HÜMNOS oder dem Global HMI geben unter anderem Funktionen zum Aufrufen, Starten, Unterbrechen und Stoppen des NC-Programms sowie einzelne Parameter zur Prozessverfolgung vor [HUEM98, MICH00]. Gemäß den Anforderungen müssen zusätzlich die Elemente der frei erweiterbaren NC-Programmierschnittstelle und die Schnittstellenfunktionen der NC-Algorithmen als Aufruffunktionen im HMI nutzbar sein. Die Lösung ist eine generische Funktionsschnittstelle innerhalb des HMI. Ausgehend von dem Windows Betriebssystem bietet sich hierzu das Windows-Messaging-System an. Demnach werden die verschiedenen HMI-Module als eigenständige Fenster implementiert und in die Rahmen der Oberfläche geladen. Mit dem Grundsystem und untereinander interagieren sie durch Versenden und Abarbeiten von Nachrichten. Jede der Nachrichten besteht aus einem Nachrichtenschlüssel, der die auszuführende Funktion kodiert, und zwei Datenblöcken für die Eingangs- und die Rückmeldedaten. Sowohl die Schlüssel als auch die Daten können auf der Grundlage von Basistypen anwenderspezifisch implementiert werden und garantieren so die geforderte Interoperabilität trotz einer umfangreichen Offenheit für Änderungen der NC-Programmierschnittstelle und dem Funktionsumfang des HMI.

7.1.6. Schnittstellenserver zur Kopplung mit der NC-Steuerung

Interface to link the HMI modules with the numerical control

Es gibt heute keine steuerungsübergreifenden, einheitlichen Schnittstellen. Jeder Steuerungshersteller legt ausgehend von eigenen oder standardisierten Softwaremechanismen die Daten und Funktionen seines Steuerungskerns offen. *Brühl* hat jedoch zeigen können, dass sich offene NC-Steuerungen um Applikationsinterfaces (API) ergänzen lassen, so dass sie nach außen eine einheitliche Schnittstelle anbieten, die er auf OSACA aufgesetzt hat [BRUE03].

Anstelle die Schnittstellen in der NC-Steuerung zu vereinheitlichen und damit ggf. nicht benötigte API Elemente rechen- und speicherintensiv vorzuhalten, wird die Schnittstelle erst im weniger zeitkritischen HMI abgebildet. Der Ansatz sieht einen Schnittstellenserver im HMI-Grundsystem vor, in dem die API-Elemente der HMI-Module und die der NC-Steuerung angemeldet und verknüpft werden. Ausgehend von einer Analyse offener Steuerungssysteme lassen sich die drei Grundelemente ableiten:

- Variablen: Austausch von Daten zwischen HMI und NC
- Events: Reaktion auf Ereignisse im HMI oder in der NC
- Prozesse: Aufruf von Funktionen im HMI oder in der NC

7.1.6.1. Variablenobjekte

Variable objects

Im Gesamtsystem müssen Einstellparameter, Steuerungsdaten und Parameter zur Prozessmanipulation innerhalb der Module des HMI und mit der NC-Steuerung ausgetauscht werden. Der Umfang der zu kommunizierenden Variablen hängt von den Funktionen der HMI- und NC-Module ab und soll sich gemäß der Anforderungsanalyse durch den Entwickler frei erweitern lassen. Der Schnittstellenserver wird dazu den OSACA Ansatz einer namensbasierten Zuordnung zwischen Server- und Klienten-Variablen angelehnt. Jedoch wird der starre Systemhochlauf von OSACA nicht übernommen, der vorgibt, dass zuerst die Servervariablen angelegt sein müssen, bevor die Klienten-Variablen initialisiert und verknüpft werden können. Das entwickelte Konzept sieht vor, dass sowohl die Server- als auch die Klienten-Variablen als eigenständige Objekte im Schnittstellenserver registriert werden. Erst zur Laufzeit werden die Schnittstellenobjekte mit den passenden Gegenstellen verbunden. Auf diese Weise kann eine Verbindung jederzeit neu konfiguriert, getrennt oder mit einem nachträglich bereitgestellten Serverobjekt verbunden werden. Nicht verbundene Klienten werden in der Systemoberfläche als Warnung angezeigt. Mit diesem Ansatz ist die Anforderung erfüllt, dass sich Funktionsmodule und deren Schnittstellen dynamisch laden und entladen lassen.

Der schnellste Datenaustausch wäre über den direkten Zugriff auf einen gemeinsamen Speicherbereich möglich. Da jedoch im HMI mehrere, voneinander unabhängig implementierte Module integriert sind, kann es beim parallelen Speicherzugriff zu Kollisionen kommen. Zudem ist gefordert, dass das Gesamtkonzept auf verteilte Systeme anwendbar ist, d.h. Variablen müssen sich speicherübergreifend austauschen lassen. Dazu bildet der Schnittstellenserver die Variablen als Datenobjekte in einer Zwischenschicht ab und koordiniert die Lese- und Schreibzugriffe über Objektmethoden. Während ein Modul den Variableninhalt neu setzt, ändert sich der Status des Objekts. Der Zugriff wird für andere Module gesperrt oder so lange verzögert, bis der Schreibvorgang abgeschlossen ist und das Datum sicher ausgelesen werden kann. Damit zeitkritische Funktionen der NC- und HMI-Module auf den Zugriffsstatus reagieren können, bietet der Schnittstellenserver priore und wartende Methoden an. Ein Zugriff mit hoher Priorität erhält Vorrang, so dass ein Regeleingriff nicht durch eine Anzeigefunktion ausgebremst wird. Die Reaktionsgeschwindigkeit wird zurückgemeldet und kann anforderungskonform in der Benutzungsoberfläche angezeigt werden. Dem Anwender wird an den Eingriffsmechanismen angezeigt, ob eine Eingriffsmethode gesperrt ist oder ob sie zeitverzögert reagiert.

Des Weiteren fasst der Schnittstellenserver die Variablenabrufe mehrerer Module zusammen. Wenn Positionsdaten gleichzeitig von der Fortschrittsanzeige des HMI als auch einer integrierten Überwachungsroutine abgefragt werden, ruft die Schnittstelle die Kommunikation zur NC-Steuerung nur einmal auf und stellt den aktualisierten Wert allen Modulen zentral zur Verfügung. Der Modulentwickler spezifiziert während der Anmeldung seiner Klientenobjekte, ob eine Variable in festen Zeitfenstern aktualisiert wird, nur bei Änderungen oder nur bei gezielten Zugriffen. Somit wird die Verbindung zur NC-Steuerung nicht unnötig belastet.

7.1.6.2. Events

Events

Um die einzelnen Module über die Änderung einer Variablen zu informieren, werden Ereignisse, sogenannte Events, eingesetzt. Die Events lösen in den Server- und Klientenobjekten Methoden aus, die wiederum Callback-Funktionen aufrufen, die auf die Funktionen der registrierten Objekte und Module zeigen. Dieser Mechanismus stellt sicher, dass die HMI- und NC-Module nicht zyklisch miteinander kommunizieren müssen, um auf den Wechsel eines Arbeitsschritts oder eine geänderte Werkzeugposition reagieren zu können. Die Funktionsaufrufe erfolgen ereignisorientiert, ohne die Kommunikationsressourcen unnötig zu belasten.

Abgesehen von den bereits beschriebenen Klientenobjekten entspricht dieser Mechanismus den OSACA Events. Somit können die Schnittstellenfunktionen einer OSACA Steuerung als Spezialisierung der Schnittstellenobjekte implementiert und als steuerungsspezifisches Kommunikationsmodul in die HMI-Plattform geladen werden.

Der Mechanismus ist notwendig, um Überwachungs- und Regelroutinen an die jeweilige Bearbeitungssituation anzupassen. Beispielsweise wird die Schnittkraftüberwachung vor dem Wechsel von einer Planoperation auf eine Zustellbewegung informiert und kann entsprechend umparametriert oder deaktiviert werden. Die Forderung von *Kaever* nach einer satzbezogenen Überwachung [KAEV04] kann so auf der Basis der NC-Programmverarbeitung und den von der NC-Steuerung bereitgestellten Variablen präzise auf die Bearbeitungsgeometrie, Verfahrwege, Operationswechsel und Maschinenfunktionen abgestimmt werden. Zusätzlich werden die Parameteranzeigen, die Bearbeitungssimulation und die Dialoge zur Prozessmanipulation über die Events aktualisiert.

7.1.6.3. Prozessobjekte

Process objects

Mithilfe der Variablenobjekte können die Module des HMI und der NC-Steuerung untereinander auf ihre offenen Datenschnittstellen zugreifen und eventgesteuert vordefinierte Funktionen zu deren Verarbeitung auslösen. Damit ist zwar indirekt ein modulübergreifender Funktionsaufruf möglich, aber es werden keine Aufrufparameter übergeben oder Ergebnisse zurückgeliefert. Diese Mechanismen eignen sich somit nicht, um die übergreifenden Funktionen der Module von außen aufzurufen.

Ausgehend von der OSACA Kommunikationsplattform definiert *Hardebusch* eine Funktionsschnittstelle als Operation, die nicht den Inhalt eines Datenobjekts, sondern durch ihre Abarbeitung den Zustand des gesamten Moduls verändert. Er beschreibt dazu einen Zustandsautomaten, den das Modul schrittweise als Ergebnis der verschiedenen Funktionsaufrufe durchläuft [HARD02]. Bezogen auf den Interpreter erlaubt ein solcher Zustandsautomat erst nach erfolgreicher Anwahl eines NC-Programms den Start der Abarbeitung, der wiederum die Voraussetzung für eine Programmunterbrechung oder den Bearbeitungsabbruch ist. Dieser Mechanismus ist insbesondere für die entwickelten Prozessregelfunktionen von Bedeutung, da hierbei verschiedenste Restriktionen berücksichtigt werden müssen. Beispielsweise darf der Werker in einer T-Nut den Nutenfräser nicht in Achsrichtung zurückziehen. Stattdessen muss ihm das Gesamtsystem situationsgerecht Funktionen anbieten, die das Werkzeug entlang der abgearbeiteten Bahn freifahren oder das Futter öffnen, um das Werkzeug lösen und die Arbeitsspindel zurückziehen zu können.

Analog zu den Datenschnittstellen werden auch die Funktionsschnittstellen in Form von Objekten gekapselt und ihr Zugriff innerhalb des Schnittstellenserver verwaltet. Sobald ein Modul gestartet wird, meldet es seine bereitgestellten und geforderten Funktionen an den Schnittstellenserver und initialisiert diese als Server- oder Klienten-Prozessobjekte. Der Begriff Prozessobjekt zeigt, dass auch hier die OSACA Kommunikationsmechanismen als Ansatz übernommen werden. Über ein Konfigurationsfile oder einen hart kodierten Namen identifiziert der Schnittstellenserver die Objekte und verbindet sie untereinander. Wird ein Server mit gleichem Namen zweimal angeboten, so erscheint eine Warnmeldung und die Plattform fordert den Benutzer auf, die Konfiguration eindeutig zu ändern. Entgegen der OSACA Plattform wird nicht vorausgesetzt, dass ein Serverprozessobjekt existiert. Ein fehlender Server wird durch Platzhalterobjekte ersetzt, die zum einen dem Anwender die fehlende Gegenstelle melden und zum anderen versuchen nach jedem neu geladenen Modul die passende Serverschnittstelle zu finden.

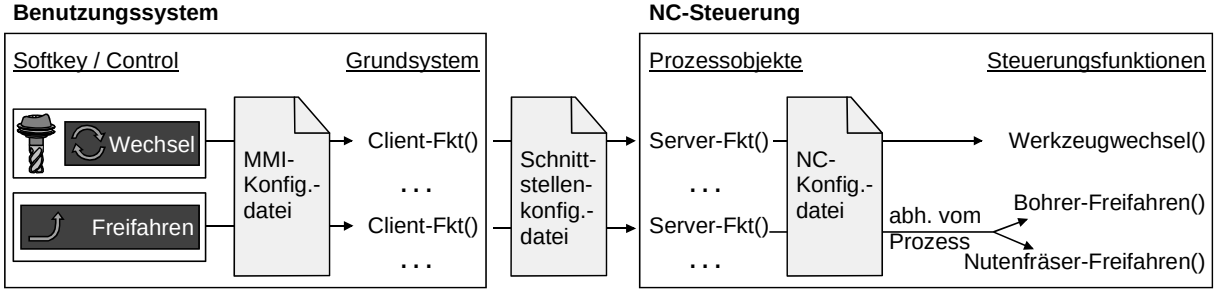


Bild 7.6: Prozessobjekte zum situationsgerechten Eingriff in die NC-Programmverarbeitung
Process objects for context sensitive reacting on manipulations in the NC-program

Ausgehend von diesen flexiblen, nicht statisch verbundenen Funktionsschnittstellen wird die Anforderung erfüllt, Eingriffsmethoden zur Manipulation der NC-Programmverarbeitung in Abhängigkeit von der aktuellen Bearbeitungsoperation bereitzustellen. Der Entwickler einer NC-Programmroutine definiert seine spezifischen Regelfunktionen und stellt diese als Funktionsschnittstelle extern zur Verfügung. Im Schnittstellenserver des HMI erkennt ein wartendes Klientobjekt die neue Funktion der NC-Steuerung, verbindet sich mit ihr und aktiviert situationsgerecht die passende Schalterbelegung und Anzeigeelemente. Beispielsweise werden mit dem Start einer Taschenoperation dem Werker die Zustellung und die Eingriffsbreite als Regelgrößen angezeigt und auf variable Overrideschalter gelegt.

7.2. Aufgabenorientierte Arbeitsweise

Task oriented workflow

Ausgehend von der entwickelten NC-Programmierschnittstelle lässt sich die Bearbeitungsaufgabe analog der Denkweise des Werkers und seinem Vorgehen bei der NC-Programmierung in Bearbeitungsmerkmale, abzuarbeitende Operationen und Maschinenfunktionen strukturieren. Die Umsetzung in ausführbare Weg- und Schaltanweisungen erfolgt entweder autonom innerhalb der NC-Steuerung oder aber der Programmierer nutzt die objektorientierte Programmstruktur, um die Abarbeitung der einzelnen Aufgaben weiter zu detaillieren und explizit vorzugeben. Entgegen der konventionellen NC-Programmierung mit G-Codes müssen die Planer und Werker von der Prozess- und Operationsplanung im CAP/CAM über die Programmkodierung bis hin zur Prozessregelung nicht zwischen verschiedenen Anwendungen und Datenformaten wechseln.

Nachfolgend werden die auf der konzipierten, zentralen Programmverwaltung aufbauenden Teilfunktionen des Benutzungskonzepts detailliert. Hierzu gehören die NC-Programmierung, die Simulation und Bearbeitungsverfolgung sowie die Prozessüberwachung und -regelung.

7.2.1. NC-Programmierung an der NC-Steuerung

NC-programming at the machine tool

Aufgrund der Vielzahl der zur Aufgaben- und Prozessbeschreibung entwickelten NC-Programmelemente und ihrer mitunter komplexen Abhängigkeiten ist es nicht zielführend, das NC-Programm textbasiert in einem Editor zu erstellen. Folglich müssen dialoggeführte Systeme zum Erstellen und Anzeigen der NC-Programme bereitgestellt werden.

Bezogen auf die Programmierung an der Werkzeugmaschine sieht das Konzept vor, dass die Funktionen zum Parsen und Schreiben des Programminhalts als ein zentraler Dienst der Benutzungsoberfläche angeboten werden. Wie in Kapitel 7.1.4 beschrieben, sind diese Funktio-

nen generisch aufgebaut. Sie können alle Elemente einer beliebig erweiterten NC-Programmierschnittstelle verwalten. Dadurch vereinfacht sich der Funktionsumfang eines zu integrierenden Programmiersystems auf das Auswerten und werkergerechte Darstellen des Programminhalts sowie die Bereitstellung von Dialogen und Auswahlmenüs, mit denen sich das NC-Programm erstellen, ergänzen oder korrigieren lässt.

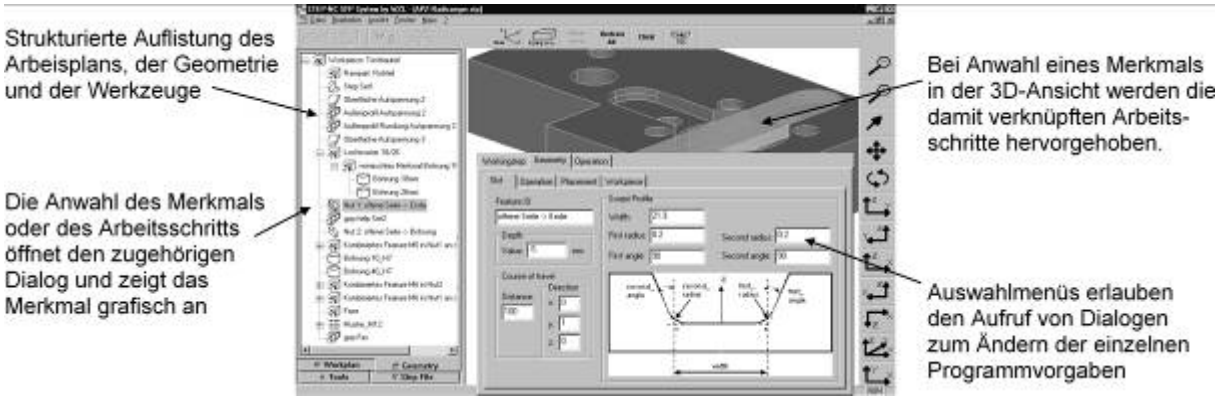

Strukturierte Auflistung des Arbeisplans, der Geometrie und der Werkzeuge
Die Anwahl des Merkmals oder des Arbeitsschritts öffnet den zugehörigen Dialog und zeigt das Merkmal grafisch an
Bei Anwahl eines Merkmals in der 3D-Ansicht werden die damit verknüpften Arbeitsschritte hervorgehoben.
Auswahlmenüs erlauben den Aufruf von Dialogen zum Ändern der einzelnen Programmvorgaben

Bild 7.7: Zugriff auf die Arbeitsschritte und spezialisierte Dialoge zur NC-Programmierung
Access to the STEP-NC workingsteps and dialogs for NC-programming

Aufgrund des objektorientierten Aufbaus der Programmierschnittstelle und der steuerungsseitigen Offenheit Arbeitsschritte, Operationen und Merkmale in Abhängigkeit von dem eigenen Prozesswissen individuell optimiert abzuarbeiten, ist es nicht zielführend, das Programmiersystem als eine geschlossene Anwendung in die Benutzungsoberfläche zu integrieren. Stattdessen bietet sich eine aufgabenorientierte Aufteilung in übergreifend gültige sowie auf einzelne Programmelemente spezialisierte Module an. Zu den übergreifend gültigen Elementen gehören beispielsweise die Abfolge der Arbeitsschritte im Arbeitsplan, der die Merkmale des Roh- und Fertigteils beschreibende Geometriebaum sowie die einzusetzenden Werkzeuge. Sie geben eine Art Leitfaden vor, der den Werker durch das NC-Programm führt und einen systematischen Zugriff auf den Programminhalt vorgibt. Ausgehend von dieser Systematik wird auf die Details der Arbeitsschritte, Merkmale und Werkzeuge verwiesen. Analog zu der für die Programminterpretation entwickelten Vorgehensweise werden die Zugriffsfunktionen nicht starr implementiert, sondern in einer Konfigurationsdatei werden die einzelnen Merkmale frei den spezialisierten Funktionsmodulen zugeordnet, Bild 7.7.

Die einzelnen Funktionsmodule werden beim Systemstart geladen und als Dialoge, Ansichten oder Schaltflächen in der Oberfläche ergänzt. Aufgrund der Vielzahl der Programmelemente werden die Dialoge zur Programmerstellung kontextabhängig geladen. Beginnend mit einem neuen NC-Programm wird zuerst eine Fertigteilgeometrie geladen oder erstellt. Diesem Fertigteil weist der Programmierer ein Rohteil sowie die abzuarbeitenden Bearbeitungsmerkmale, wie Flächen, Taschen, Konturen etc., zu. Um den Umfang der programmierbaren Merkmale flexibel erweitern zu können, werden Auswahllisten angeboten, die alle in der Programmierschnittstelle enthaltenen und durch Dialoge in der Benutzeroberfläche erstellbaren Elemente enthalten, Bild 7.8. Diese Liste ist Teil des HMI-Grundsystems. Die darin enthaltenen Elemente werden zum einen als Grundlage für die Zuordnung in der beschriebenen Konfigurationsdatei genutzt und zum anderen leiten sie sich aus der Vererbungsstruktur der Elemente ab. So wird beispielsweise sichergestellt, dass als Geometriemerkmale nur die in der Programmierschnittstelle auch enthaltenen Elemente zur Geometrieprogrammierung angeboten werden.

Der gleiche Ansatz wird auf die Programmierung des Arbeitsplans mit den darin enthaltenen Arbeitsschritten und NC-Funktionen übertragen. Beginnend mit dem Projekt wählt der Programmierer schrittweise die im Ablauf zu ergänzenden Elemente aus. Bei der Anwahl eines

Elementes bietet das Grundsystem erneut eine generische Maske an, in der ausgewählt werden kann, ob ein neues Element hinzugefügt oder ein vorhandenes ausgewählt werden soll. Wenn in die Benutzeroberfläche eine grafische Darstellung als Modul geladen wurde, so wird über die Anwahl eines Arbeitsschritts das Merkmal automatisch hervorgehoben. Dazu meldet das Grundsystem lediglich die Merkmalreferenz an die geladenen Anzeigen, die dann entsprechend ihrer Implementierung reagieren.

Bild 7.8: Dialoggeführtes Erstellen und Ändern der Bearbeitungsgeometrie
Dialogs for generating and editing machining features

Die zentrale Programmverwaltung verfolgt im Hintergrund, welche Programmvorgaben noch nicht bearbeitet oder unvollständig gesetzt sind. Bevor das Programm abgeschlossen werden kann, warnt sie den Programmierer und bietet ihm an die entsprechenden Programmiermodule aufzurufen. Auch hierbei wird der Mechanismus genutzt, dass die geladenen Dialoge jeweils einem Element der Programmierschnittstelle zugeordnet sind.

Sind zu einem Element der NC-Programmierschnittstelle keine Module, d.h. Dialoge oder Anzeigen, konfiguriert, so greift das Benutzungssystem auf generische Eingabemasken zurück, die das Setzten der Programmelemente ermöglichen, gleichzeitig aber auch auf die fehlende spezifische Funktionalität hinweisen.

Alle Planungsdaten vom Werkstück mit den zu bearbeitenden Merkmalen über den Arbeitsplan, die Spannlagen bis hin zu Operationen, Strategien und Werkzeugen werden schrittweise dialoggeführt eingegeben und im NC-Programm abgelegt. Da dieses Datenformat auch von der NC-Steuerung gelesen und verarbeitet werden kann, entfällt der konventionell folgende Schritt eines Postprozessorlaufs. Die Planungsdaten werden verlustfrei und durchgängig von der Planung über das WOP-System bis in die NC-Steuerung genutzt.

7.2.2. Simulation und Bearbeitungsverfolgung

Simulating and tracking the machining process

In kommerziellen Systemen ist die Simulation Teil der CAP/CAM Planungssoftware oder aber direkt in die WOP-Tools integriert. Diese Systeme interpretieren das NC-Programm eigenständig und stellen seinen Inhalt grafisch dar. Sie sind nicht direkt mit der NC-Steuerung

gekoppelt und zeigen folglich anstelle des realen einen virtuellen, über die Ausführungszeit und das NC-Programm synchronisierten Bearbeitungsfortschritt an [OPEN04, SIEM02, HOFF05]. Auch die Simulation des Steuerungs- und Maschinenverhaltens, durch einen virtuellen NC-Kern in Kombination mit einem Maschinenmodell, erlaubt nur eine offline Betrachtung des geplanten Bearbeitungsverlaufs [KINZL06]. Abweichungen, die sich im Prozess aufgrund der Werkzeugkorrektur oder aber nach Eingriffen durch den Anwender ergeben, können sie nicht vorhersehen und daher nicht simulieren. Den aufgestellten Forderungen nach einem gemeinsamen Programmzugriff durch mehrere Anwendungen und die NC-Steuerung, einer Simulation der real ausgeführten Programmabarbeitung und der Möglichkeit eigene Bearbeitungsmerkmale sowie Algorithmen zur Bahnauflösung zu integrieren, entsprechen diese Systeme nicht.

Simulation im Vorfeld der Bearbeitung

Die konzipierte Beareitungssimulation setzt keine eigenen Algorithmen zur Berechnung der Werkzeugbewegungen ein. Sie greift auf die Werkstück- und Geometriebeschreibung des NC-Programms zu und überlagert hierzu die in den NC-Workingsteps der NC-Steuerung berechneten Verfahrwege, Bild 7.9. Dazu stellt die NC-Steuerung ihre Bahnauflösung und Geometrieverarbeitung analog dem von Siemens entwickelten virtuellen NC-Kern [MENZ05] der Oberfläche zur Verfügung. Die Anbindung kann wahlweise über das direkte Auslesen von Sollpositionen erfolgen oder aber die Steuerung liefert zu jeder abgearbeiteten Operation die berechneten Werkzeugbewegungen, d.h. den *Toolpath* zurück. Gegenüber den Positionsangaben hat der Toolpath den Vorteil, dass er unabhängig von der Bahngeschwindigkeit und dem Interpolationstakt ist. Er beschreibt die Werkzeugbahn in Form von Polylinien, Kreisbögen oder Spline beliebig genau.

Indem jederzeit der Bezug zwischen der simulierten Werkzeugbewegung und dem abzuarbeitenden Merkmal hergestellt werden kann, vereinfacht sich die Kollisionsbetrachtung. Innerhalb der Benutzeroberfläche kann ein Modul die aktuelle Merkmalsgeometrie mit den aus der Steuerung gelesenen Bahnen verrechnen und die Schnittpunkte als Kollision anzeigen.

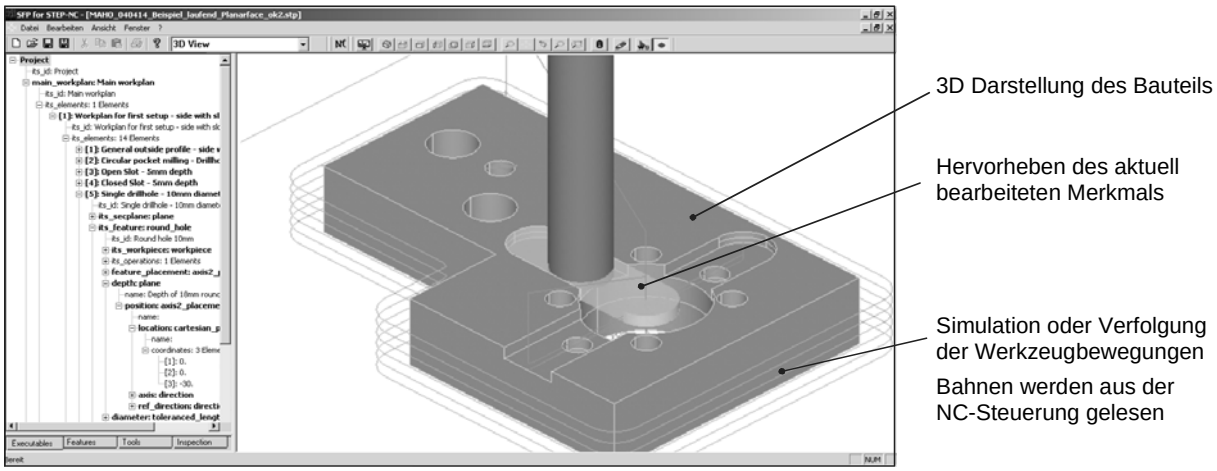

3D Darstellung des Bauteils
Hervorheben des aktuell bearbeiteten Merkmals
Simulation oder Verfolgung der Werkzeugbewegungen
Bahnen werden aus der NC-Steuerung gelesen

Bild 7.9: Einheitliche Anzeige zur Bauteildarstellung, Simulation und Prozessverfolgung
Common view to display the part, the simulation and the machining progress

Da die NC-Programmierschnittstelle STEP-konform konzipiert ist, lassen sich die relevanten Flächen und Merkmale aus einer STEP Konstruktionsdatei ohne Konvertierungsverluste in das NC-Programm übernehmen. Somit wird der heute zum Erkennen von Kollisionen und Restmaterial notwendige Vergleich einer separaten CAD Datei mit den G-Codes des NC-Programms umgangen (vgl. Vericut der Fa. CGTech [CGTE04]). Nach einer Änderung am CAD-Modell oder den Prozess- und Operationsdaten im NC-Programm wird die Simulation

über die in Kapitel 7.1.4.2 beschriebene Programmverwaltung und Synchronisationsmechanismen automatisch aktualisiert. Dieser Ansatz stellt sicher, dass alle in die Benutzeroberfläche integrierten Funktionen simultan aktiviert werden. Die Aufgabenstellung, den Werker umfassend bei der Programmerstellung, -überprüfung und -optimierung zu unterstützen, wird erfüllt.

Verfolgung der laufenden Bearbeitung

Bezogen auf die Abarbeitung in der NC-Steuerung ist einer der wesentlichen Vorteile des entwickelten Datenmodells, dass die Fortschrittsverfolgung nicht mehr auf die textbasierte Anzeige elementarer Steuerungsanweisungen in der Benutzungsoberfläche beschränkt ist. Stattdessen lassen sich dem Werker technologische Zusammenhänge für ihn verständlich, grafisch und dialoggeführt darstellen. Insbesondere bei einer durch die Bauteilgröße, Späne oder Kühlschmierstoffe eingeschränkten Sicht auf die Bearbeitungsstelle hilft es dem Werker, wenn ihm das aktuell zu bearbeitende Merkmal und die Werkzeugbewegung parallel zur Bearbeitung oder im Vorlauf angezeigt werden.

Zur Umsetzung dieser Anforderung bietet es sich an, die zuvor beschriebene Simulation mit dem realen Prozess sowie den im Vorlauf der Programmverarbeitung gespeicherten NC-Workingsteps abzugleichen. Hierzu meldet die NC-Steuerung den aktuell in Bearbeitung befindlichen Arbeitsschritt an die Benutzeroberfläche. Da beide Teilsysteme auf das gleiche NC-Programm zugreifen, ermittelt die Programmverwaltung des Benutzungssystems das dazugehörige Merkmal und meldet diese Informationen an die verbundenen MMI-Funktionen. Im Arbeitsplan wird daraufhin der aktive Arbeitsschritt gekennzeichnet und die Simulation hebt die zu bearbeitende Geometrie hervor, Bild 7.9. Indem darüber hinaus die Programmverarbeitung der NC-Steuerung die NC-Programmelemente nutzt, um die generierten Bewegungsbahnen an die Simulation zu melden, werden dem Werker die abgearbeiteten sowie die noch ausstehenden Verfahrbewegungen angezeigt. Aufgrund der durchgängigen Verfolgung der NC-Workingsteps über die Geometrieverarbeitung der NC-Steuerung bis hin zur Sollwertausführung, lassen sich zudem die Ist-Positionsdaten gezielt mit Verweis auf den STEP-NC *Workingstep* und das aktuelle Bahnsegment an die Oberfläche melden. Aus der Simulation der geplanten Roh- und Fertigteilkontur mit den steuerungsseitig generierten Bearbeitungsbahnen wird durch die Überlagerung der Ist-Bewegung eine realistische Darstellung der aktuellen Bearbeitung.

Softwarekonzept einer offen erweiterbaren grafischen Darstellung

Vergleicht man die Implementierung heutiger 3D Geometriedarstellungen, so sind die zugrundeliegenden Grafikbibliotheken vergleichbar aufgebaut. Bibliotheken, wie Open Cascade [CASC07] oder das World Toolkit [SENS07] bieten vordefinierte Funktionen zum Erstellen von Flächen- und Volumendarstellungen an. Diese werden über boolesche Operationen addiert oder voneinander abgezogen, um die verschiedenen Werkstück- und Bearbeitungsgeometrien darzustellen. Ferner ist es mithilfe sogenannter Frames oder Knoten möglich, die modellierten Teilgeometrien zusammenzufassen und in eine hierarchische Struktur einzuordnen. Zu jeder derart strukturierten Geometrie werden eine eigene Koordinatenverschiebung und Darstellungseigenschaften, wie die Farbe oder die Transparenz definiert.

Bei der NC-Programmierung oder während der Bearbeitungssimulation können somit gezielt Merkmale herausgehoben oder ausgeblendet werden. Die Koordinatenverschiebungen zwischen den Knoten ermöglicht es, das Roh- und Fertigteil analog der im NC-Programm enthaltenen Beschreibung aufzubauen. Ausgehend von dem Werkstückträger werden die Spannlagen der einzelnen Werkstücke und dazu wiederum die Bearbeitungsmerkmale positioniert. Während der Bahnsimulation werden die von der Steuerung gemeldeten Verfahrwege über diese Koordinatensysteme der Sollgeometrie im Paletten-, Werkstück- oder Merkmalskoordinatensystem überlagert.

Werden darüber hinaus die Knoten offen gelegt, so kann ein externer Entwickler für seine individuellen Werkzeuge und Merkmale die grafische Darstellung ergänzen. Im Idealfall würde die grafische Simulation zu jedem Bearbeitungsmerkmal die individuelle Zuordnung eines Darstellungsalgorithmus als Erweiterung der Grafikbibliothek zulassen.

Zusatzinformationen an den Knoten oder aber eine Liste, in der die Knoten den NC-Programmelementen zugeordnet werden, erlaubt die bidirektionale Interaktion mit der grafischen Darstellung. Meldet die Programmverwaltung die Bearbeitung an einem spezifischen Merkmal, so wird dieses einem Knoten zugeordnet und die damit verbundenen Geometrien werden optisch hervorgehoben. In der Gegenrichtung selektiert der Werker über die Maus ein Merkmal und wählt für dieses Merkmal gezielt eine der MMI-Funktionen zur NC-Programmierung, Prozessüberwachung oder -manipulation auf. Dies wäre über die Wegsegmente des G-Code und deren fehlendem Bezug zur Fertigungsgeometrie nicht möglich.

7.2.3. Prozessüberwachung und –regelung

Process monitoring and control

Indem die NC-Steuerung den aktuellen Arbeitsschritt in der NC-Programmverwaltung über ein Statusbit markiert oder aber als Datenobjekt direkt an die Benutzeroberfläche meldet, besteht dort die Möglichkeit die Anzeigen, Dialoge und Funktionen kontextabhängig zu schalten. Innerhalb des Konzepts wird dieser Mechanismus genutzt, um gezielt Regler zu aktivieren und deaktivieren, Programmierdialoge oder Teile davon zu sperren oder Überwachungsfunktionen aufzurufen und zu parametrieren.

Ausgehend von der vollständigen Aufgabenbeschreibung, werden die Eingriffs- und Überwachungsfunktionen gezielt für einzelne Arbeitsschritte und Teiloperationen aktiviert. In einer Baum- oder Listendarstellung des Arbeitsplans kann der Werker den Bearbeitungsfortschritt verfolgen. Die einzelnen Regler, Dialoge und Überwachungsfenster werden zwischen den Arbeitsschritten deaktiviert, operationsgerecht gewechselt und wieder aktiviert. Hierzu bieten sich die Zeiten an, während denen das Werkzeug nicht im Eingriff ist, d.h. positioniert oder gewechselt wird.

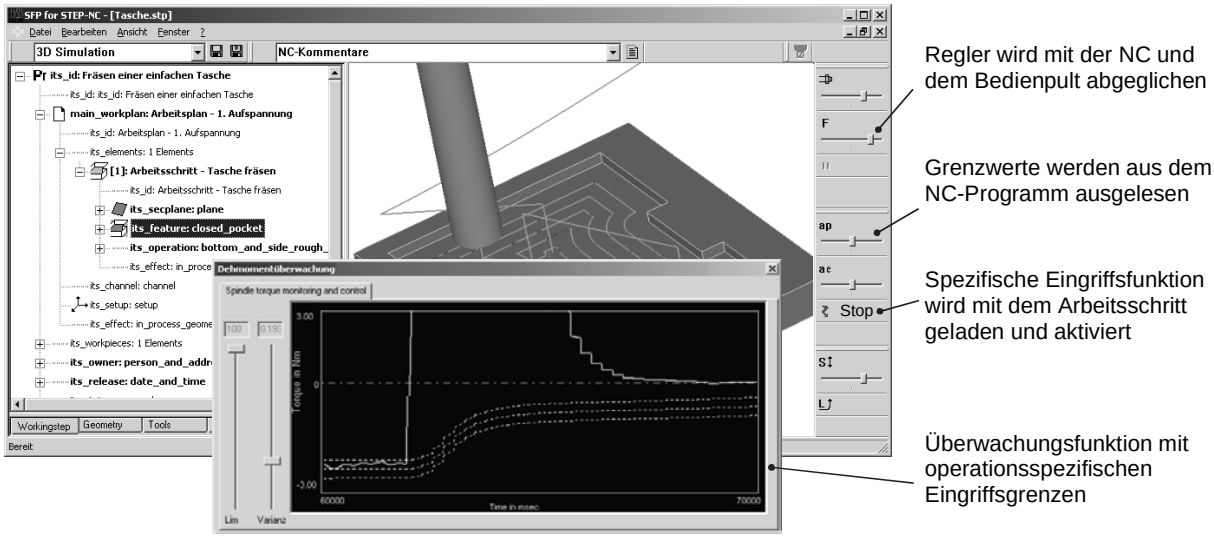


Bild 7.10: Aktivieren der Prozessüberwachung und -regelung in Abhängigkeit der Operation

Activating the process monitoring and control based on the active operation

Bei schnell aufeinander folgenden Arbeitsschritten muss die Benutzeroberfläche dem Werker die Überwachungs- und Eingriffsfunktionen frühzeitig bereitstellen. Anderenfalls kann er

zum einen nicht im voraus Parameter korrigieren und er findet die einzelnen Regler, Ansichten und Dialogelemente nicht schnell genug. Eine mögliche Lösung besteht darin, die Ansicht der Oberfläche zu unterteilen und die einzelnen Segmente anlog der Abarbeitungsfolge mit zu belegen. Gemäß den Anforderungen wird der Anwender entsprechend den ihm aus der Planung bekannten Arbeitsschritten, Operationen und Merkmalen unterstützt. Er kann den Prozess aufgabenorientiert verfolgen und bearbeitungsparallel korrigieren.

8. Realisierung einer NC-programmbasierten Prozessregelung

Realization of a process control based on a task and feature oriented NC-program

Die entwickelten Konzepte und Algorithmen wurden im Rahmen des Sonderforschungsbereiches 368 „Autonome Produktionszelle“ [APZ02, PFEI06] und den durch die Europäische Union geförderten STEP-NC Forschungsprojekten [STEP01, STEP04] in mehreren Prototypen umgesetzt. Hierzu gehören die Anbindung an CAM-Systeme, die Implementierung von Postprozessoren zur Abarbeitung der entwickelten NC-Programmierschnittstelle in konventionellen NC-Steuerungen sowie die Realisierung einer vollständig integrierten Steuerungslösung, die den Werker von der NC-Programmierung über die Fortschrittsverfolgung bis zur Prozessmanipulation aufgaben-, merkmals- und operationsorientiert unterstützt. Alle Systeme wurden an praktischen Bearbeitungsbeispielen demonstriert und die Ergebnisse mit Forschungs- und Industriepartnern evaluiert [WEYR00, WECK03a , STEP04].

Die im Rahmen dieser Arbeit entwickelte NC-Programmierschnittstelle, wurde für die Fräs- und Drehbearbeitung international abgeglichen und als ISO Standard veröffentlicht [ISO 14649-1]. Da diese Modelle bei der ISO als Standard oder als Normungsentwürfe vorliegen, werden ihre Programmelemente im Folgenden nicht explizit beschrieben. Im Anhang sind Auszüge des in den vorangegangenen Kapiteln beschriebenen Datenmodells abgebildet.

Das Kapitel 8.1 gibt einen Überblick über die Funktionen der realisierten Prototypen und deren softwaretechnischen, flexibel erweiterbaren Aufbau. Daran anschließend wird in Kapitel 8.2 exemplarisch die adaptive Programmverarbeitung anhand der für das Ausfräsen einer Tasche implementierten NC-Workingsteps erklärt. Abschließend werden in Kapitel 8.3 die praktische Anwendung und die dabei gemessenen Reaktionszeiten dargestellt.

8.1. Aufbau und Teilfunktionen der realisierten Prototypen

Structure and sub-functions of the realized prototypes

Zur Realisierung und Bewertung der steuerungsintegrierten, adaptiven Programm- und Bahnauflösung reichte es nicht aus, nur die Steuerungsmodule und deren Algorithmen zu erweitern. Aufgrund der entwickelten, umfangreichen NC-Programmierschnittstelle mussten zusätzlich Systeme zum Erstellen, Anzeigen und Manipulieren der Programme implementiert werden. Ein wesentliches Ziel war es, sowohl die NC-Programmierschnittstelle als auch die erweiterten steuerungsseitigen Funktionen darstellen und erproben zu können. Die im Rahmen dieser Arbeit realisierten Prototypen sind in Bild 8.1 dargestellt.

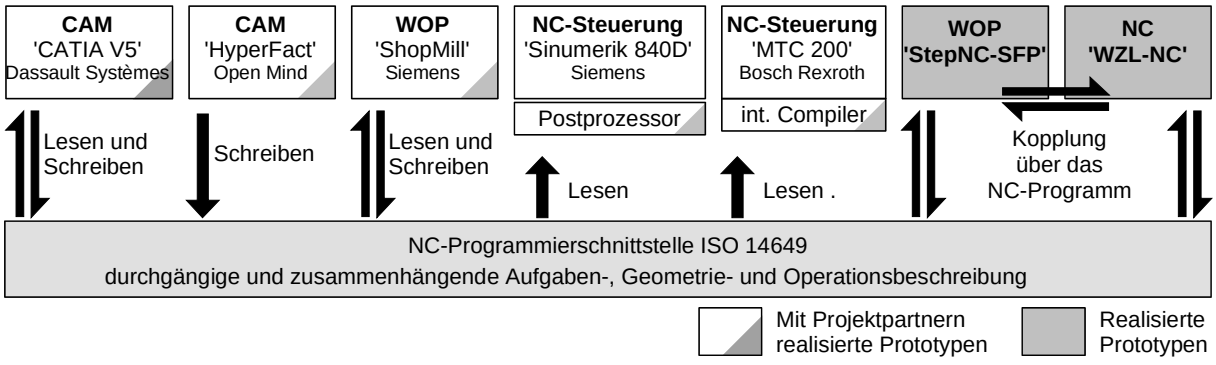


Bild 8.1: Überblick über die im Rahmen dieser Arbeit (mit-)entwickelten Systeme
Overview to the systems (co-)realized within this thesis

8.1.1. Zugriff auf die NC-Programmierschnittstelle

Access to the NC-programming interface

Das in den vorangegangenen Kapiteln erarbeitete Konzept trennt eindeutig die im NC-Programm gespeicherte Beschreibung der Bearbeitungsaufgabe und deren Umsetzung auf Sollwerte und elementare Steuerungsfunktionen durch die NC-Steuerung. Aufgrund der Vielzahl der möglichen Fertigungsprozesse, Maschinenkonfigurationen und Umsetzungsstrategien ist eine statische Einbindung der NC-Programmierschnittstelle sowie eine fest kodierte Abbildung der Programmelemente auf einzelne Funktionen nicht zielführend. Entsprechend dem entwickelten Konzept wurde daher eine Hilfsanwendung implementiert, die das Datenmodell der NC-Programmierschnittstelle generisch auf eine Klassenbibliothek abbildet und einheitliche Zugriffsfunktionen zum Lesen und Schreiben des Programminhaltes bereitstellt. Da die Funktionen der Bibliothek nur von den Datentypen und nicht von deren Inhalt oder deren Bedeutung abhängen, dient die Bibliothek ausschließlich als Schnittstelle zum Zugriff auf das NC-Programm. Erst innerhalb der Anwendungen werden die spezifischen Programmelemente erkannt und in spezialisierten Funktionsmodulen weiterverarbeitet, Bild 8.2.

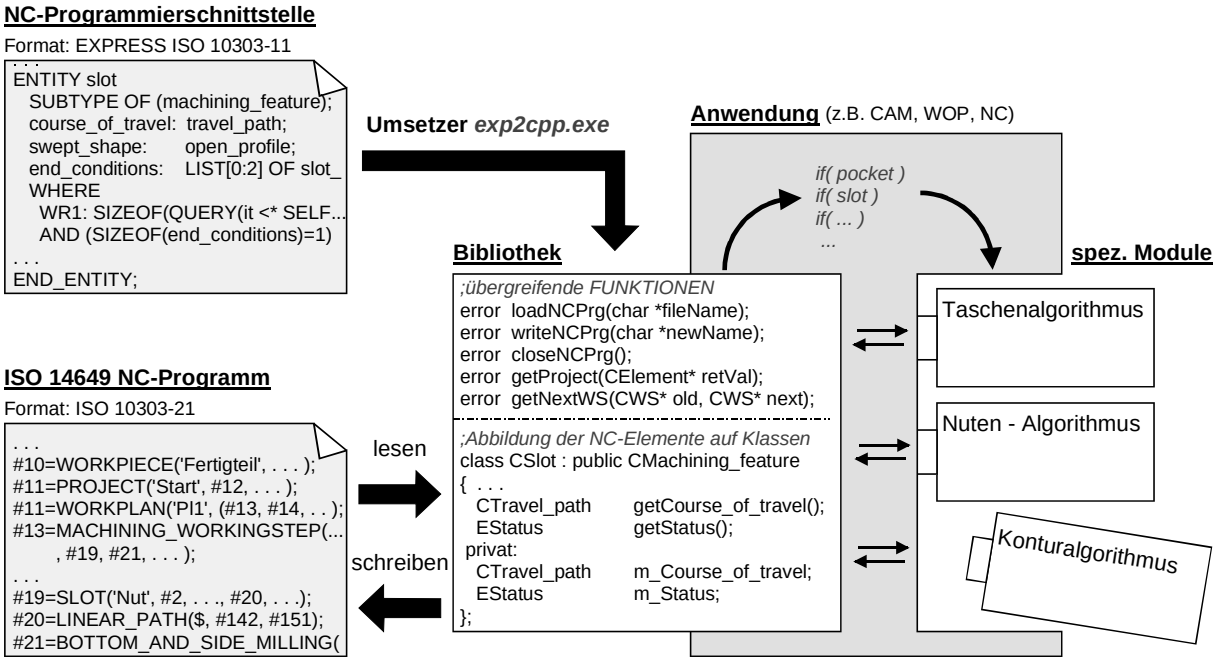


Bild 8.2: Generisch generierte Bibliothek zum Zugriff auf das STEP-NC-Programm

Generic library to access the STEP-NC-program

Die Abbildung des in EXPRESS beschriebenen Datenmodells auf eine C++ Klassenbibliothek baut auf dem am WZL von *Schulz* und *Rosenberg* entwickelten *exp2cpp-Umsetzer* auf [SCHO00]. Entgegen kommerziell verfügbaren Toolkits bot sich diese Lösung an, da die so entwickelten Prototypen von allen universitären und industriellen Projektpartnern ohne Lizenzen genutzt werden können. Zudem lässt sich der *exp2cpp-Umsetzer* offen erweitern, um die für das Datenmodell der ISO 14649 typischen Grundfunktionen automatisch zu generieren und zentral in der Bibliothek bereitzustellen. Beispielsweise lädt und schreibt die Bibliothek das NC-Programm. Zudem sucht sie aktiv nach dem Programmstart „*Project*“ und erlaubt von dort aus den gezielten Zugriff auf den Arbeitsplan, die darin enthaltenen Arbeitsschritte und deren Merkmale und Operationen. Diese Grundfunktionen erleichterten die Entwicklung der realisierten Programmier- und Steuerungssysteme. Der Entwickler kann sich auf die Umset-

zung einzelner Arbeitsschritte konzentrieren, ohne dass er die gesamte NC-Programmierschnittstelle betrachten muss. Somit entspricht die Bibliothek dem konzipierten, modularen Softwareaufbau, dessen Funktionen sich spezifisch erweitern oder durch eigene, optimierte Routinen überladen lassen, Bild 8.2 links.

Der Umsetzer generiert zu jedem Programmelement eine Klasse mit Zugriffsfunktionen und internen Statusattributen. Ausgehend von den in der NC-Programmierschnittstelle vorgegebenen Regeln, prüfen die Objekte der Klasse in ihren Methoden selbstständig, ob ihre Parameter vollständig und innerhalb der vorgegebenen Grenzwerte gesetzt sind. Fehlerhafte Eingaben werden abgelehnt. Offene Definitionen werden gekennzeichnet, so dass der Programmierer informiert oder die Abarbeitung durch die Steuerung rechtzeitig unterbrochen werden kann. In der Steuerung werden die Statusattribute zudem genutzt, um anzuzeigen, ob ein Element für Eingriffe gesperrt werden soll, es aktuell abgearbeitet wird oder bereits abgearbeitet ist.

Um ein fortlaufendes Abfragen und Überprüfen der Programmelemente auf Änderungen zu verhindern generiert der Umsetzer Funktionslisten innerhalb der elementspezifischen Klassen. In diesen Listen hinterlegen die Systemanwendungen Rückruffunktionen. Diese werden automatisch aufgerufen, wenn sich der Status oder der Inhalt eines Objektes ändert. Sie reagieren damit ereignisgesteuert und werden zur Anzeige des Abarbeitungsfortschritts oder zur Parametrierung der Prozessüberwachung und -regelung genutzt.

Ferner kann die Bibliothek von mehreren Anwendungen und Modulen parallel eingebunden werden. In ihrem Speicherbereich wird das NC-Programm zentral geladen und gemeinsam ohne redundante Kopien gemeinsam genutzt. Änderungen an den Programmelementen stehen unmittelbar systemweit zur Verfügung. Die einzelnen Elemente lassen sich über die Funktionen der Bibliothek neu erstellen, löschen, kopieren, manipulieren und als Nachrichten intern austauschen. Diese Lösung dient als Grundlage, um beispielsweise in der Programmieroberfläche eine Änderung durch mehrere Simulations- und Überwachungsfunktionen zu überprüfen, bevor sie in das Programm übernommen wird.

Die Funktionen der Bibliothek enthalten keine prozess-, merkmals- und operationsspezifische Logik. Sie haben primär die Aufgabe, auf das physikalische NC-Programm zuzugreifen und dessen Inhalt den verarbeitenden Funktionen der Programmier- und Steuerungssysteme bereitzustellen. Die logische Auswertung und das Ableiten von Reaktionen übernehmen die Programmier-, Anzeige- und Steuerungssysteme.

8.1.2. CAM-Systeme zur ISO 14649 Programmerstellung

CAM System for generating ISO 14640 NC-programs

Aufgrund der Vielzahl der Datenelemente und ihrer komplexen Referenzierung innerhalb des NC-Programms führt eine manuelle Programmerstellung zu Fehlern und ist nicht praktikabel. Ferner kann sie nicht die Anwendbarkeit sowie die Durchgängigkeit der Programmierschnittstelle von der Planung bis in die NC-Steuerung demonstrieren. Im Rahmen dieser Arbeit wurden daher in Zusammenarbeit mit industriellen Projektpartnern die CAM-Systeme HyperFACT von OpenMind und CATIA V5 von Dassault Systèmes um die Ein- und Ausgabe ISO 14649 konformer NC-Programme erweitert, Bild 8.1.

Beide Lösungen binden die zuvor beschriebene Bibliothek als Schnittstelle ein, um auf den Inhalt der NC-Programme zuzugreifen und neue Programme zu erstellen. Dazu verwaltet die Bibliothek die im NC-Programm enthaltenen Programmelemente und deren Referenzen, generiert neue Elemente und löscht sie auf Anforderung mitsamt ihren Abhängigkeiten. Die Umsetzung innerhalb der CAM-Systeme vereinfacht sich dadurch auf das Aufrufen, Auslesen sowie das Befüllen der Programmelemente und ihrer Parameter.

Der Vergleich der in den Eingabemasken der beiden CAM-Systeme abgefragten Parameter mit den Elementen der NC-Programmierschnittstelle zeigt, dass sich deren Datenmodelle größtenteils direkt auf die Schnittstelle abbilden lassen. Diese Eigenschaft resultiert aus der an die Entwicklung der ISO 14649 gestellten Anforderung, das NC-Programm durchgängig und umfassend von der Planung bis in die Fertigung einsetzen zu können. Ausgenommen sind nur optimierte, herstellerspezifische Operationen und Strategien, die jedoch technisch durch eine Erweiterung der NC-Programmierschnittstelle und deren generische Umsetzung in die vorgestellte Bibliothek realisiert werden können. Auf dieser Grundlage vereinfacht sich die Implementierung der Prototypen auf die Zuordnung, das sogenannte Mapping der einzelnen Datenelemente. Die bestehenden Dialoge, Planungs- und Detaillierungsroutinen, Simulationen und Anzeigen der Systeme sind für den Anwender weiterhin voll nutzbar. Zusätzlich erlaubt ihm die neu eingebundene NC-Programmierschnittstelle den Austausch der Planungs- und Programmvorgaben zwischen den verschiedenen CAM-Systemen als auch den nachfolgend realisierten WOP-Systemen und NC-Steuerungen, Bild 8.3.

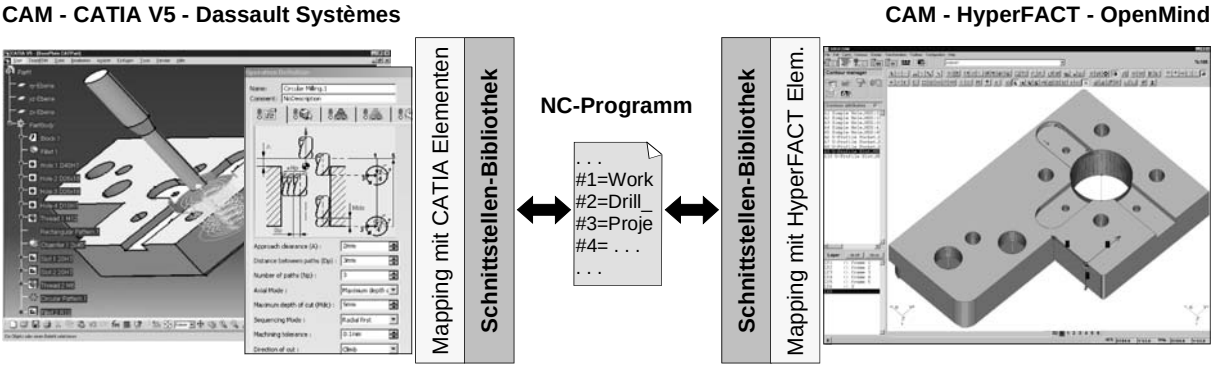


Bild 8.3: Zur Ein- und Ausgabe der ISO 14649 erweiterte kommerzielle CAM-Systeme
Commercial CAM systems enabled to read and write ISO 14640 NC-programs

Der Umfang der realisierten Elemente entspricht der in der ISO 14649 für die prismatische Fräsbearbeitung definierten Konformitätsklasse 2 (1 = einfache Bahnen, 2 = Bearbeitungsmerkmale, …, 6 = zusätzlich Flächen- & Topologieinformationen). Mithilfe der angebundenen CAM-Systeme lassen sich NC-Programme zur Bearbeitung von Ebenen, Konturen, Nuten, Bohrungen und Taschen planen und schrittweise detaillieren. Zudem wurde die Programmierung von gruppierten Merkmalen und Wiederholmustern implementiert.

8.1.3. Steuerungsanbindung über WOP-Systeme

Pre-processing ISO 14649 to G-Codes with shop-floor-oriented programming systems

Analog zu der Programmerstellung in der Arbeitsvorbereitung ist auch an der Maschine eine softwaretechnische Unterstützung notwendig, damit der Werker den umfangreichen Inhalt des NC-Programms nachvollziehen und ändern kann. Hierzu wurde in Zusammenarbeit mit Siemens deren WOP-System *ShopMill* zur Umsetzung der entwickelten Programmierschnittstelle erweitert und gleichzeitig als Postprozessor zur Anbindung der *Sinumerik 840D* eingesetzt.

Bei der Einbindung von ShopMill werden die Merkmale und Operationen des NC-Programms auf die, für die Siemens NC-Steuerung *Sinumerik 840D* spezifischen und innerhalb von ShopMill anzeig- und editierbaren NC-Zyklen abgebildet. Analog zu den beschriebenen CAM-Systemen wird wiederum die entwickelte Bibliothek als Schnittstelle zwischen den Programmdaten und der logischen Umsetzung eingesetzt, Bild 8.4.

Die einzelnen Merkmale und die daran auszuführenden Operationen geben die interne Softwarestruktur des Umsetzers vor. Sie werden von einer zentralen, Arbeitsschritte abrufenden

Softwarefunktion erkannt und von ihr spezialisierten Funktionsmodulen zur Abbildung auf die passenden Zyklen und Programmstrukturen zugewiesen. Aufgrund der mächtigen Zyklenpakete des WOP-Systems ShopMill müssen dazu keine neuen Algorithmen zur Abarbeitung der in der ISO 14649 für die Fräsbearbeitung definierten Bearbeitungsmerkmale entwickelt und implementiert werden.

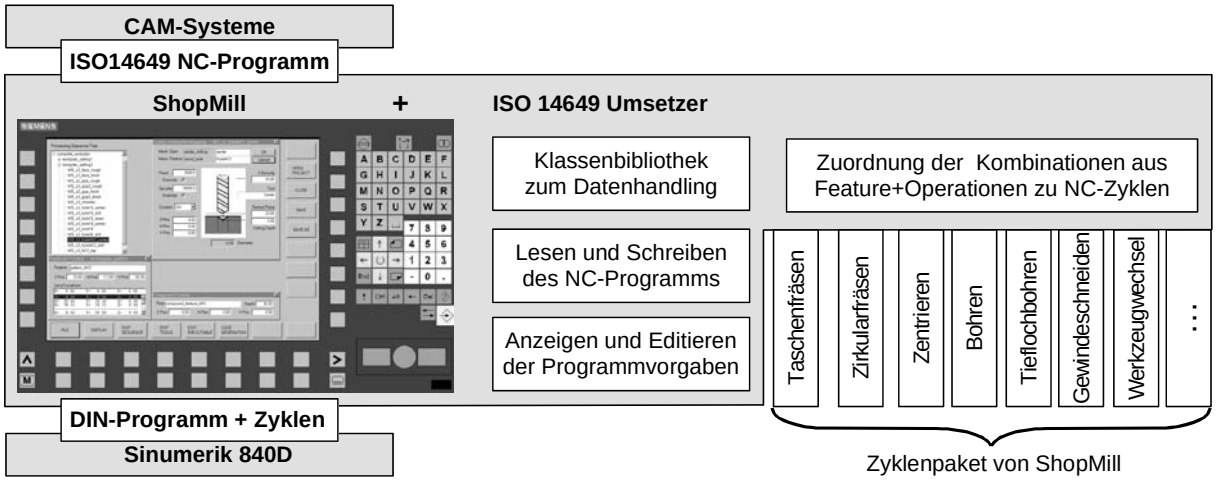


Bild 8.4: Integration der *Sinumerik 840D* über ShopMill und NC-Zyklen
Integration of the Sinumerik 840D with ShopMill and NC-cylces

Einzig die Werkzeugdefinitionen, Nullpunktverschiebungen und Verbindungsbewegungen werden von zusätzlichen Routinen ermittelt und als einfacherer G-Code in das Siemens spezifische NC-Programm geschrieben. Ausgehend von dieser Realisierung war die *Sinumerik 840D* die erste STEP-NC fähige NC-Steuerung und wurde zur Erprobung unterschiedlicher Frässzenarien genutzt [STEP02] und für die Drehbearbeitung erweitert [HEUS03]. Dies zeigt, dass die Umsetzung auf Zyklen und G-Codes einen effektiven und einfach realisierbaren Migrationsschritt vom G-Code hin zur Programmierung nach ISO 14649 darstellt. Jedoch kann der Umsetzer die vollständige Planungsinformation nicht bis in die Programmabarbeitung fortführen und dem Werker eine aufgaben- und operationsorientierte Prozesskorrektur während der Bearbeitung ermöglichen.

8.1.4. Statische NC-Programmumsetzung im Interpreter

Static mapping of the NC-Program in the interpreter

Um dem Anwender die in der Planung definierten Arbeitsschritte, Merkmale und Operationen parallel zur Bearbeitung anzeigen zu können, wurde für die NC-Steuerung *MTC 21* von Bosch Rexroth die Umsetzung der ISO 14649 Programme in deren Interpreter integriert. Der Umweg über einen externen Umsetzer entfällt. Der Werker ruft das NC-Programm direkt auf.

Die Interpretation der Programmvorgaben, und deren Abbildung auf die für die *MTC* spezifischen, elementaren Steuerungsbefehle erfolgt in einem sogenannten Compiler. In Bild 8.5 sind der Compiler und seine internen Funktionsmodule dargestellt. Nach der Programmanwahl wird der Compiler gestartet ❶. Er lädt zuerst Konfigurationsdateien ❷, in denen der Steuerungshersteller oder ein Entwickler definiert, welche spezialisierten DLLs zur Verarbeitung der verschiedenen Merkmale und Operationen geladen werden sollen. Ferner werden die in der Steuerung gespeicherten Werkzeuge und Nullpunktverschiebungen ausgelesen ❸. Die Ablaufinterpretation ❹ steuert das weitere Vorgehen, indem sie den Inhalt des NC-Programms über die beschriebene Bibliothek lädt ❺ und schrittweise die darin enthaltenen Arbeitsschritte abarbeitet ❻. Innerhalb der Arbeitsschritte werden die Typen der program-

mierten Merkmale, Operationen und NC-Funktionen interpretiert zur Verarbeitung an die konfigurierten NC-Workingsteps übergeben ❼. Diese lösen die Programmvorgaben schrittweise in Sub-Workingsteplisten auf und geben diese an die Ablaufinterpretation zurück. Von dort werden sie über das steuerungsspezifische Interface an die nachfolgenden Module des NC-Kerns weitergeleitet.

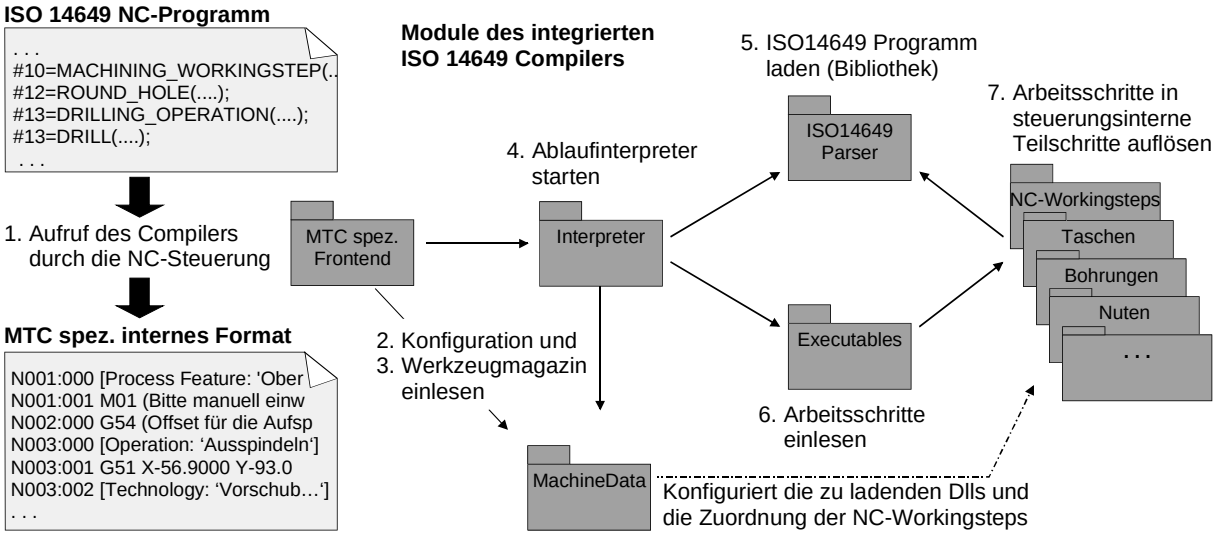


Bild 8.5: Integrierte Programmverarbeitung in der *MTC 21* von Bosch Rexroth
Integrated processing of the NC-program in the Bosch Rexroth MTC 21

Im Fall der MTC wird der Compiler nicht über den Abarbeitungsfortschritt gesteuert. Daher werden alle NC-Workingsteps sofort ausgeführt, d.h. sie generieren die für die nachfolgenden Funktionen des NC-Kerns notwendigen Eingangsinformationen. Im Fall der MTC sind das spezifische Steuerungsbefehle, G-Codes und Anzeigeinformationen, die in der internen Dateiverwaltung der MTC über Textdateien ausgetauscht werden, Bild 8.5 unten links. Mithilfe der Anzeigeinformationen wird dem Werker während der Bearbeitung der Programmfortschritt zusätzlich zu den Befehlssätzen angezeigt. Sie bleiben als Kommentare in der Steuerungsoberfläche sichtbar und stellen den Bezug zu dem Arbeitsplan sowie den abzuarbeitenden Merkmalen und Operationen her. Der Werker kann somit die Vorteile der entwickelten NC-Programmierschnittstelle an der Steuerung nutzen und die Abarbeitung anhand der für ihn verständlichen Aufgaben- und Merkmalsbeschreibung verfolgen. Eingriffe in die laufende Abarbeitung sind jedoch aufgrund der generierten G-Codes nur über die konventionellen Mechanismen, wie die Overrides möglich.

8.1.5. Regelbare NC-Programmumsetzung in der NC-Steuerung

Manipulatable processing of the NC-program in the numeric control

Mit dem Ziel, die in Kapitel 8.1.4 generierten NC-Workingsteps auch in den nachfolgenden Steuerungsmodulen des NC-Kern fortzuführen, wird der realisierte Compiler als eigenständiger Interpreter in die institutseigene NC-Steuerung *WZL-NC* eingebaut. Zudem werden die internen Kommunikationsstrukturen erweitert, so dass der Bezug zu den Programmvorgaben und den NC-Workingsteps bis an die Sollwertschnittstellen der Steuerung erhalten bleibt. Auf dieser Grundlage werden die konzipierten Eingriffsfunktionen, d.h. der Zugriff des Werkers in die Programmvorgaben parallel zur Bearbeitung realisiert.

Die WZL-NC ist eine modular aufgebaute und verteil auf mehreren Hardwareplattformen implementierte NC-Steuerung für die 5-achsige Fräs- und Laserbearbeitung. Sie wurde entwickelt, um über eine vorausschauende, splinebasierte Geschwindigkeitsführung Freiformflä-

chen abzuarbeiten [FAUS97, SCHO00]. Erste Algorithmen zum Abarbeiten von rechteckigen Ebenen, Taschen und Bohrungen wurden von *Brouër* implementiert [BROU00]. Da der Quellcode aller Steuerungsmodule, vollständig offen liegt und objektorientiert in C++ programmiert wurde, bietet sich die *WZL-NC* als Plattform zur durchgängigen Realisierung des erarbeiteten Konzeptes von der Benutzeroberfläche bis in die Prozessregelung an.

Die Hardware der *WZL-NC* besteht aus zwei Rechnern: Die Geometriekette und die softwarebasierte SPS laufen auf einem VME-Bus System der Fa. Motorola unter dem Echtzeitbetriebssystem VxWorks. Die Benutzerschnittstelle, die Werkzeugverwaltung sowie der Interpreter sind auf einem Industrie-PC mit Windows NT installiert. Innerhalb eines Rechners kommunizieren die Module über OSACA-Kommunikationsmechanismen basierend auf Shared Memory. Zwischen den beiden Hardwareplattformen wird die OSACA-Kommunikation über Ethernet und das TCP/IP Protokoll aufgebaut. Dieser Aufbau wurde übernommen, jedoch mussten die Benutzerschnittstelle und der Interpreter vollständig neu implementiert werden, um im laufenden Prozess den NC-Programminhalt adaptieren und durchgängig zur Kommunikation nutzen zu können. Darüber hinaus wurde die Geometriekette erweitert, um den Bezug zu den NC-Workingsteps des Interpreters sowie Korrekturstrategien bis an die Sollwertschnittstellen fortzuführen, Bild 8.6.

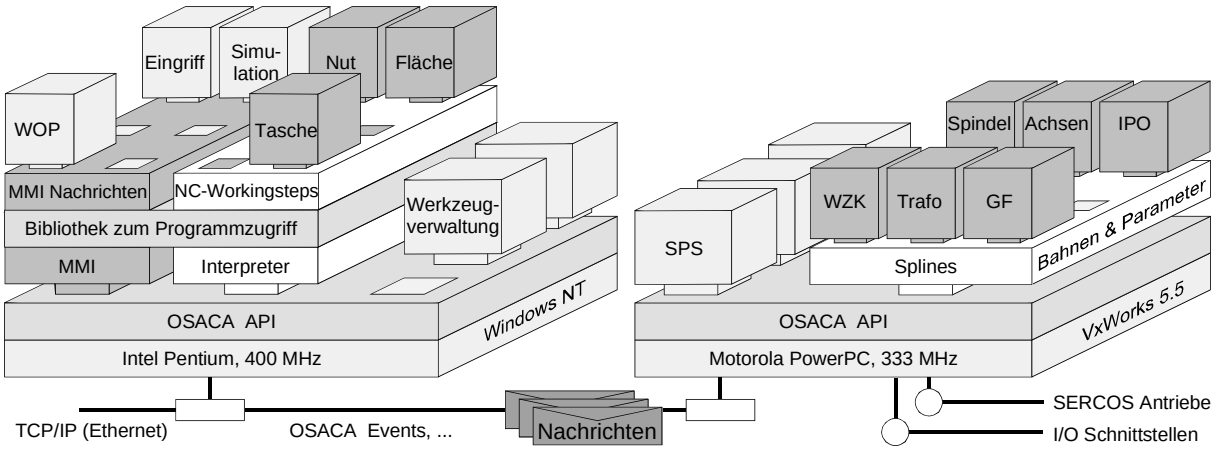


Bild 8.6: Aufbau der WZL-NC und der realisierten Erweiterungen
Structure of the WZL-NC and the realized extensions

Gegenüber dem erarbeiteten Konzept, das eine schnelle Umsetzung von Programmeingriffen durch die Adaption der NC-Workingsteps in einem von allen Steuerungsmodulen gemeinsamen genutzten Arbeitsspeicher vorsieht, erfordert der verteilte Aufbau der *WZL-NC* ein aufwändigeres Kommunikationskonzept. Die Module der *WZL-NC* tauschen untereinander ihre Ergebnisse über OSACA Kommunikationsmechanismen (Variable, Prozessobjekte) sowie zwei unterschiedlich priorisierte Nachrichtenketten aus. Daher setzt die Realisierung auf dem in Kapitel 6.3 für verteilte Hardwareplattformen vorgeschlagene Ansatz modulspezifischer Nachrichten auf. Die im Interpreter generierten NC-Workingsteps werden an einem *WZL-NC* spezifischen Frontend (vgl. Kapitel 8.1.4) in steuerungsinterne Nachrichten aufgelöst.

Entgegen den vorherigen statischen Steuerungs-Workingsteps der *WZL-NC* verfolgen die neuen NC-Workingsteps die Nachrichten und deren weitere Abarbeitung über Statusflags und Rückmeldungen aus den einzelnen Steuerungsmodulen. Erst nach dem vollständigen Abarbeiten eines NC-Workingsteps und der von ihm generierten Nachrichten stellt dieser seine Überwachung ein und wird gelöscht. Bis dahin aktualisiert er fortlaufend sein internes Prozessmodell, um im Fall eines Eingriffs von diesem aktuellen Punkt aus neue Bahnen und Schaltbefehle zu generieren.

Die nicht deterministische Ethernet-Kommunikation der *WZL-NC* wird über im Interpreter präventiv generierte Korrekturvektoren kompensiert. Die Vektoren werden in der Geometriekette auf die Maschinenkinematik abgebildet und über die Spline der Achsbewegung parametriert. Im Eingriffsfall verrechnet der echtzeitfähige Interpolator die Vektoren anteilig mit den Sollbahnen. So wird zum einen ein stabiler Prozesspunkt bestimmt und zum anderen die Zeit überbrückt, bis der Interpreter über die Kommunikationskette adaptierte Bahnen bereitstellt.

Die für den Compiler beschriebenen Konfigurationsmechanismen (Kapitel 8.1.4) werden für den Interpreter sowie das MMI übernommen. Beim Systemstart, als auch dynamisch nach der Anwahl eines neuen NC-Programms laden die beiden Systeme die konfigurierten DLLs und passen ihre interne Funktionszuordnung an. Ferner binden die Systeme die entwickelte Bibliothek ein, um parallel auf das NC-Programm zuzugreifen, den Abarbeitungsfortschritt über die Programmelemente abzugleichen und ausgeführte Bahnen oder Eingriffe auszutauschen.

Die für das MMI entwickelte Programmieroberfläche setzt sich aus mehreren Funktionsmodulen zusammen, die als DLLs entsprechend der Konfigurationsdatei geladen und den Programmelementen zugeordnet werden, Bild 8.7. Die Anzeigen werden sowohl zur Programmerstellung, -anzeige als auch zur Fortschrittsverfolgung genutzt. Hierzu wird in den Darstellungen und Dialogen die objekt- und ablauforientierte Struktur des NC-Programms genutzt. Durch den Werker oder den gekoppelten Interpreter werden die aktiven Elemente angewählt und hervorgehoben. In der laufenden Bearbeitung werden die Überwachungs- und Eingriffsschnittstellen automatisch aktiviert. Hierzu meldet der Interpreter über die Elemente der NC-Programmierschnittstelle den aktiven Arbeitsschritt. Der Interpolator und die Transformation liefern über ihre OSACA-Schnittstellen fortlaufend die Lage-Ist-Werte. Diese werden im MMI protokolliert und simuliert. Ferner werden die Module des NC-Kern offline aufgerufen, um vor der späteren Abarbeitung die Bewegungsbahnen zu berechnen und in der Oberfläche gegenüber den programmierten Merkmalen auf Kollisionen zu überprüfen.

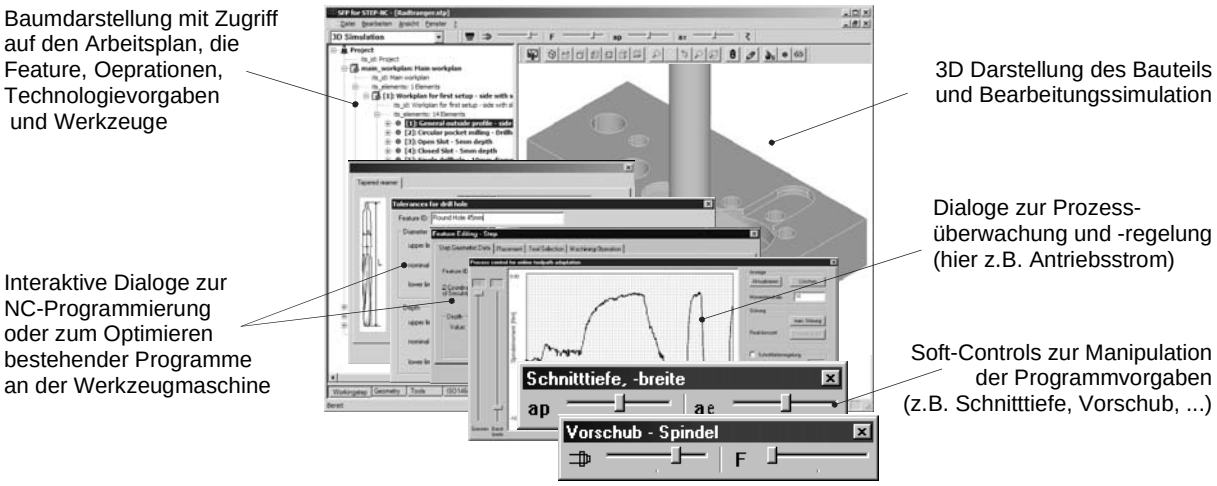

Baumdarstellung mit Zugriff auf den Arbeitsplan, die Feature, Oepratione, Technologievorgaben und Werkzeuge
3D Darstellung des Bauteils und Bearbeitungssimulation
Dialoge zur Prozess-überwachung und -regelung (hier z.B. Antriebsstrom)
Interaktive Dialoge zur NC-Programmierung oder zum Optimieren bestehender Programme an der Werkzeugmaschine
Soft-Controls zur Manipulation der Programmvorgaben (z.B. Schnitttiefe, Vorschub, ...)
Schnitttiefe, -breite
ap
ae
Vorschub - Spindel
F

Bild 8.7: MMI Funktionen zur NC-Programmierung, Prozessüberwachung und –regelung
HMI functionalities for NC-programming, monitoring and control

Im Nachfolgenden Kapitel wird die Funktion des Gesamtsystems am Beispiel der Abarbeitung einer nach ISO 14649 programmierten Taschenbearbeitung detaillierter beschrieben.

8.2. Adaptive Prozessregelung am Beispiel der Taschenbearbeitung

Adaptive control realized for milling pockets

Zur Abarbeitung von Bohrungen und Nuten werden NC-Workingsteps mit parametrierten Bewegungsmustern implementiert. Nach einem Eingriff passen sie die Streckenlängen, bzw. die Endpunkte der abzufahrenden Bahnen an. Die Sub-Workingsteps übernehmen die geänderten Endpunkte als ihre neuen Startpunkte. Die Zustelltiefe adaptiert ein spezialisierter NC-Workingstep für diese Merkmale. Er wiederholt seine Sub-Workingstepliste so lange, bis dass die Zieltiefe erreicht wird. Diese adaptiven NC-Workingsteps werden sowohl für die Bearbeitung von einfachen Nuten, Außenkonturen und Tieflochbohrungen als auch für die Abarbeitung der einzelnen Zustellebenen einer Taschenbearbeitung eingesetzt.

Innerhalb der Zustellebenen einer Tasche oder Fläche ist die adaptive Bahnauflösung aufgrund der sie begrenzenden Konturen und Inseln komplexer. Hierzu werden in Abhängigkeit von den Bahnstrategien spezialisierte NC-Workingsteps realisiert.

Bei einer unidirektionalen Bahnstrategie generiert der hierfür entwickelte NC-Workingstep entlang der programmierten Richtung Bearbeitungsbahnen, deren Anfang und Ende jeweils durch Schnittpunktbestimmung mit der Merkmalskontur und ihren Inseln resultieren. Die einzelnen Bahnen werden entweder direkt oder durch zusätzlich eingefügte Abhebe-, Positionier- und Zustellbewegungen verbunden. Zur Adaption der Überdeckung wird im laufenden Prozess das Werkzeug in Richtung der vorhergehenden Bahn (kleinere Überdeckung) oder der nachfolgenden Bahn (größere Überdeckung) verschoben. Indem die adaptierte Bahn nur durch ein Verschieben in den Bereich zwischen zwei Bahnsegmente entsteht, werden Kollisionen mit der Endkontur vermieden. Nach dem Ende der laufenden Richtungsbahn löscht der Interpreter die ausstehenden Segmente und ersetzt sie durch eine neu generierte, die geänderten Bahnparameter berücksichtigende Bahnführung, Bild 8.8.

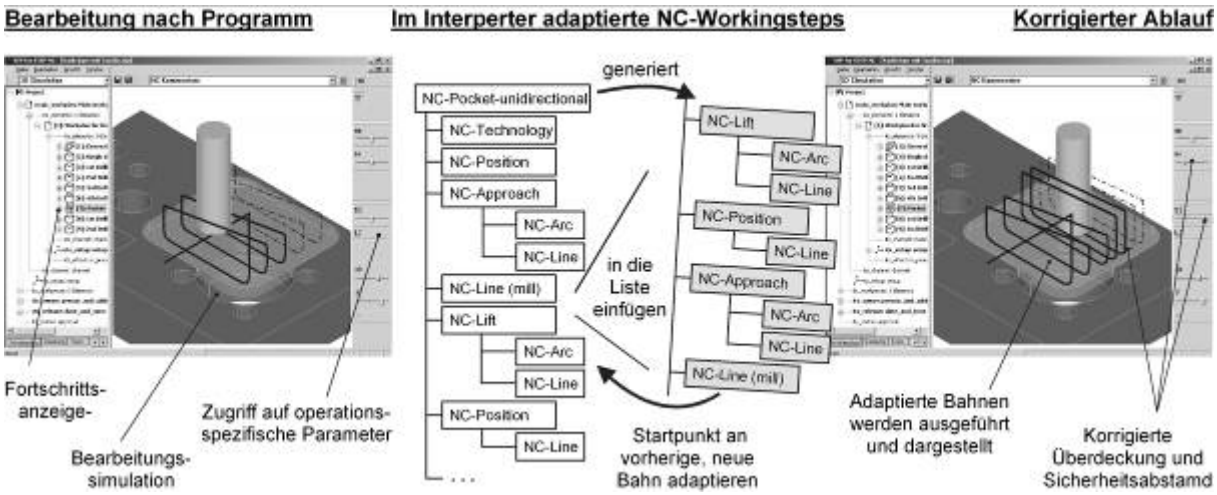


Bild 8.8: Eingriff über die Benutzerschnittstelle und Adaption der NC-Workingsteplisten

A manipulation at the HMI causes an adaptation of the NC-workingstep-lists

Für die konturparallele Schnittaufteilung müssen äquidistante Bahnen generiert und zusätzlich um tangentiale Verbindungsbewegungen ergänzt werden. Zur Bestimmung der Äquidistanten wird ein von Held vorgeschlagener Algorithmus eingesetzt [HELD91]. Dieser berechnet ausgehend von der das Merkmal beschreibenden Kontur sogenannte Bisektoren, deren einzelne Kurvenpunkte von zwei angrenzenden Kontursegmenten gleich weit entfernt liegen. Die Summe alle Bisektoren ergibt ein Voronoi-Diagramm, über das die parallel verschobenen Kontursegmente gekürzt und asymptotische Bahnen berechnet werden, Bild 8.9.

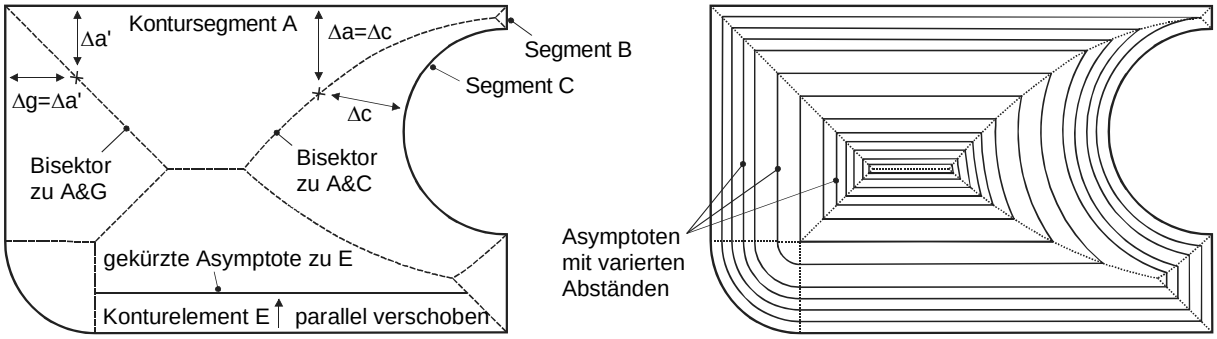


Bild 8.9: Bisektoren zur Bestimmung asymptotischer Bearbeitungsbahnen mit der Möglichkeit im Prozess die Überdeckung zur Prozesskorrektur variieren zu können

Bi-sectors used to compute asymptotic toolpaths which's distance can be varied in order to adapt the process by changing the toolpaths' overlapping

Die asymptotischen Bahnen werden nicht als statische Bahn-Workingsteps gespeichert. Sie werden runden- und ebenenweise in Sub-Workingsteps strukturiert und erhalten Zugriff auf das Voronoi-Diagramm sowie die ihnen benachbarten Asymptoten. Ferner werden zu den generierten Linearbewegungen und Kreisbogen Richtungsvektoren ausgegeben, in die das Werkzeug axial zugestellt werden kann. Diese, für eine schnelle Regelung an den Sollwertschnittstellen erforderlichen Zusatzinformationen werden von den nachfolgenden Modulen des NC-Kern auf das *WZL-NC* interne Spline-Format übertragen und entsprechend der Maschinenkinematik transformiert. Sie dienen später zur linearen Schnitttiefenkorrektur an den Sollwertschnittstellen und überbrücken die Kommunikations- und Rechenzeiten zwischen dem Interpreter und dem Interpolator.

Die einzelnen Runden-NC-Workingsteps übernehmen aus der vorherigen Runde die Endposition und generieren selbstständig Sub-Workingsteps, um Verbindungsbewegungen über die Sicherheitsebene oder tangential in der Bearbeitung zu generieren. Nach einem Eingriff werden die bestehenden Bahnen parallel um den Korrekturwert verschoben und an den Bisektoren getrimmt. Fehlende Bahnen werden in den Ebenen erkannt und ergänzt. Zudem sind die Ebenen-NC-Woringsteps so aufgebaut, dass sie ihre Bahnauflösung von einem extern vorgegebenen Startpunkt aus beginnen können. Dieser Lösung wird eingesetzt, um den im weiterlaufenden Interpolator angesteuerten Synchronisationspunkt zu übernehmen.

Im Regelfall verhalten sich diese Sub-Workingsteps als Bahnsegmente in Form von Geraden, Kreisbögen, Spline oder parametrierten An-, Abfahr- sowie Verbindungsbewegungen. Da sie eine Spezialisierung der in der Basisbibliothek bereitgestellten Datenelemente zur Wegbeschreibung darstellen, setzt sie das Hauptprogramm des Interpreters in einfache steuerungsspezifische Weginformationen, hier Geraden, Kreise und Polynomspline um. Die nachfolgenden Module der *WZL-NC* puffern die Bahnen und verarbeiten sie schrittweise zu Sollwerten für die Antriebe und regeln deren Ausführung. Innerhalb des Interpreters bleiben die Datenelemente in ihrer spezialisierten Form jedoch erhalten, bis das Segment und die übergeordnete Segmentgruppierung durch Fortschrittsmeldungen der Steuerung ihre vollständige Abarbeitung durch die Achsen gemeldet haben.

Kommt es vor diesem Abschluss zu einer Störung, so leitet das Hauptprogramm des Interpreters die Störung an die in Abarbeitung befindlichen Datenelemente der spezialisierten Bibliothek weiter. Die Anzahl der Datenelemente ist aufgrund der fortlaufenden Aktualisierung bereits minimiert, so dass die Position für eine Umplanung basierend auf den mit der Störung gemeldeten Parametern schnell gefunden und umgeplant werden kann. Bei der Umplanung kommen zwei Mechanismen zum Tragen.

8.2.1. Adaption der Schnitttiefe

Adapting the cutting depth

Nach einem Eingriff in die Schnitttiefenvorgabe wird zuerst die Offsetverrechnung des Interpolators aktiviert. Aus den über die Bearbeitungsbahn parametrierten Korrekturvektoren kann er unabhängig von einer neuen Schnittaufteilung das Werkzeug schrittweise zurückziehen oder zustellen.

Parallel werden die für die Abarbeitung des Merkmals im Interpreter generierten und zwischengespeicherten NC-Workingsteps mithilfe von OSACA-Prozessobjekten zur Adaption aufgerufen. Die geforderte Schnitttiefenkorrektur wird an die NC-Workingstepliste übertragen und in den einzelnen NC-Workingsteps mit der ursprünglichen Werkzeugzustellung verrechnet, d.h. die Koordinaten der Werkzeugachse werden adaptiert. In einem letzten Schritt wird das Höhenprofil der Bahnen mit dem Volumen des Bearbeitungsmerkmals verglichen und je nach Bedarf passt der NC-Taschen-Workingstep die letzten Bearbeitungsebenen an und fügt ergänzende Zwischen- oder Abschlussebenen in seine verkette Sub-Workingstepliste ein. Der Algorithmus steckt dabei in den einzelnen Datenelementen der Ebenen und Bahnsegmente, so dass sich diese selbständig und rekursiv adaptieren. In Verbindung mit einer Verlagerung der Berechnung in individuelle Tasks (Multitasking) wird die Umplanung maximal beschleunigt (Vgl. Kapitel 6.1.3). Die alten, nicht adaptierten Bahnsegmente, die durch die Kommunikationsplattform der *WZL-NC* gepuffert auf ihre Verarbeitung in den einzelnen Modulen warten, werden in einem letzten Schritt gelöscht. Hierzu wird ein eindeutiger Nummernschlüssel genutzt, den jedes durch den Interpreter erzeugte Segment und jede Funktion eindeutig kennzeichnet und deren Status der Interpreter über den durch die Steuerungsrückmeldung initiierten Fortschrittszeiger nach hält.

Da der Interpolator zwischenzeitlich über Offsets die alte Bahn axial zugestellt hat, kann er nahtlos mit dem Eintreffen der adaptierten NC-Workingstep-Nachrichten auf die korrigierte Bahn wechseln.

8.2.2. Adaption der Werkzeugüberdeckung

Adapting the radial overlap of the tool

Die Geschwindigkeit der Adaption wird maßgeblich von den Puffern und der Rechenzeit der einzelnen Steuerungsmodule bestimmt. Eine schnelle Reaktion, wie bei der Schnitttiefenadaption, wäre nur durch eine umfassende Erweiterung der einzelnen Steuerungsmodule und damit einer Spezialisierung der Steuerung möglich. Dieses widerspräche dem Konzept einer Basissteuerung mit modularen Erweiterungen.

Ein möglicher Lösungsansatz ist die Berechnung eines Ausweichvektors im Interpreter. Dieser wird im Interpreter lotrecht zu Werkzeugbahn und Werkzeugachse weg von der nächstgelegenen Begrenzungskontur des Beareitungsmerkmals berechnet. Die Länge des Vektors bestimmt sich aus der, aktuellen Überdeckung, dem Werkzeugdurchmesser sowie dem Abstand zu den Konturelementen. Die Transformation bildet die über die Bearbeitungsbahn parametrierten Vektoren auf die Kinematik der Maschine ab. Die Adaption wird dann im Interpolator durch Addition des Vektors zu der Sollkontur umgesetzt. Dabei wird der Offset-Betrag entsprechend der kinematischen Reserven der schwächsten Achse bestimmt. Die Rückführung in den Interpreter erfolgt wie oben beschrieben, um die nachfolgenden Bahnen anzugleichen und zu erweitern. Der Funktionsnachweis wurde basierend auf dem internen Bahnformat der WZL-NC, bestehend aus einer beliebigen Anzahl quintischer Splines, parametriert über die Weglänge, erbracht. Der Ausweichvektor wurde als zusätzlicher Spline bis zur Transformation durchgeführt und dort in vereinfachender Beschränkung auf eine 3-Achs-Kinematik ohne die beiden rotatorischen Bewegungsachsen der Versuchsmaschine umgesetzt. Das Ergebnis

waren Splines für einen Ausgleich der radialen Zustellung, die im Interpolator den Sollpositionen der zugehörigen linearen Maschinenachsen aufaddiert wurden.

Die Aufgabenstellung, eine Adaption der Überdeckung durch minimale Eingriffe in die Steuerung zu realisieren, wurde im Demonstrator ausschließlich durch Anpassungen im Interpreter realisiert. Hierzu wurde die auf die Bahnauflösung von Flächen, Taschen und Konturen spezialisierte Interpeter-Bibliothek erweitert. Damit entfiel die Notwendigkeit anwendungsspezifischer Sonderfunktionen in den kinematischen Algorithmen der dem Interpreter nach geschalteten Steuerungsmodule. Analog zur Korrektur der axialen Zustellung wird dem Interpreter eine Störung mit der umzusetzenden Parameteranpassung gemeldet. Ausgehend von der aktuellen asymptotischen Bearbeitungsrunde werden mit Hilfe der bereits verfügbaren Bisektoren alle nachfolgenden Runden mit einer korrigierten radialen Zustellung neu berechnet. Den Synchronisationspunkt für den Wechsel auf diese neue Bahn muss der Interpreter nun abschätzen: Da das Werkzeug möglichst nicht angehalten werden soll und seine Bahn nicht vorab durch andere Module korrigiert wurde, muss der alte Werkzeugweg entsprechend der Bahngeschwindigkeit sowie der Totzeit bis zur Bereitstellung der neuen Bahn im Interpolator fortgesetzt werden. D.h. die Steuerung bestimmt aus der Position der Störung und einer Verzögerungszeit, die aus der Rechenzeit des Interpreters sowie der Verarbeitungszeit der Geometriekette resultiert, den Synchronisationspunkt. Ausgehend von diesem Punkt wird im Bahnalgorithmus ein tangentialer Übergang zwischen der alten und neuen Bahn durch das Einfügen von zwei Kreisbögen realisiert. Somit entstehen am Bauteil keine Absätze durch Anrückbewegungen. Ebenso wie bei der axialen Adaption muss auch hier die Ausführung der alten, nicht adaptierten Bahnsegmente verhindert werden, was durch die beschriebene Erweiterung der Kommunikationsplattform der WZL-NC gewährleistet wurde.

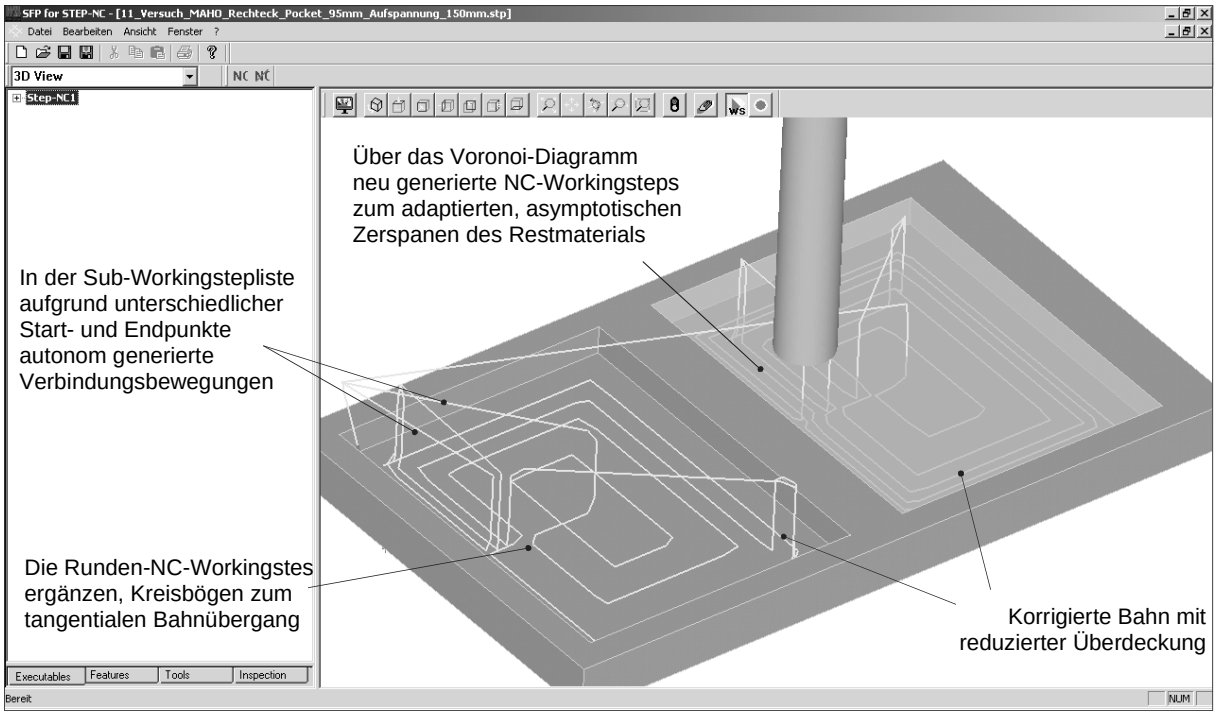


Bild 8.10: Adaption der Überdeckung bei der konturparallelen Bearbeitung

Adapting the radial overlap of a contour parallel machining operation

Die spezialisierte Datenstruktur des Algorithmus stellt somit eine optimiere Grundlage für die schnelle prozessbedingte Bahnumplanung bereit. Nach dem Abschluss der Bearbeitung des Merkmals wird die Datenstruktur einschließlich ihrer offenen, zur schnellen Reaktion im Hintergrund mitlaufenden Tasks geschlossen. In Abhängigkeit von dem nachfolgenden Bearbei-

tungsmerkmal und der hierzu geladenen spezialisierten Interpreter-Bibliothek wird dann das Prozedere erneut angestoßen.

8.3. Praktische Anwendung und Evaluation

Practical tests and evaluation

8.3.1. Austauschbarkeit der Aufgaben-, Merkmals- und Operationsdaten

Exchange of the workingsteps, features and operations based on the NC-program

Die realisierten Prototypen wurden praktisch mit den Industriepartnern Daimler, Volvo, Dassault Systèmes, Siemens und Bosch Rexroth erprobt. Das Ziel, die für den Programmierer und Werker verständlichen Aufgaben-, Merkmals- und Operationsbeschreibungen von der Planung bis an in die NC-Steuerung fortzuführen, wurde vom CAM-System über das WOP-System bis zur Fortschrittsverfolgung und Prozessregelung demonstriert.

Insbesondere bei der Erstellung und Korrektur der NC-Programme zeigten sich im Gegensatz zu der konventionellen G-Code-Programmierung Vorteile. Einmal definierte Operationen konnten einfach auf gleiche oder ähnliche Merkmale übertragen werden, da sie als Objekt innerhalb des NC-Programms gekapselt sind. Optimierungen konnten somit zentral vorgenommen werden und übertrugen sich aufgrund der Referenzierung innerhalb des NC-Programms auf die entsprechenden Arbeitsschritte und Merkmale. Der Anhang 11.2 enthält ein detailliertes Programmbeispiel.

Weiterhin wurden die Vorteile des Austauschs vollständiger Merkmals- und Operationsdaten beim Austausch der gleichen NC-Programme zwischen unterschiedlichen NC-Steuerungen und Fräsmaschinen untersucht, Tabelle 8.1. Dabei zeigte sich, dass die Programme direkt austauschbar waren. Lediglich die maschinenspezifischen Rüstinformationen mussten noch mithilfe der realisierten WOP-Systeme angepasst werden, so dass ausgehend von den Nullpunkten und realen Werkzeugen die Bearbeitung gestartet wurde. Die werkzeugangepasste Schnittaufteilung sowie die maschinenspezifische Funktionsanpassung übernahmen die NC-Steuerungen basierend auf ihren Zyklen oder den für die NC-Workingsteps realisierten Algorithmen. Der Werker an der Maschine musste keine weiteren Anpassungsarbeiten leisten und wurde ausgehend von seinem aufgaben-, merkmals- und operationsorientierten Prozessverständnis effektiv an der Maschine unterstützt.

Steuerung	Maschine	ISO 14649 Anbindung
Bosch Rexroth MTC	3-achsiger MAHO Mahomat	Compiler im Interpreter
Sinumerik 840D	5-achsige Hermle C 800 U	Postprozessor & ShopMill
	3-achsige Chiron Fz 22 S	
	5-achsige Chiron Fz 15 S	
WZL-NC	5-achsige MAHO MH 600E	Durchgängige Lösung im NCK

Tabelle 8.1: Für praktische Versuche eingesetzte NC-Steuerungen und Fräsmaschinen

Numeric controls and machine tools used for testing

8.3.2. Zeitverhalten in einer modularen, dezentralen Steuerungsarchitektur

Während der Reaktionszeit, die zwischen einem Regeleingriff und dem Wechsel auf eine von den NC-Workingsteps neu berechneten Werkzeugbahn vergeht, wird die alte Bahn fortge-

setzt, oder über eine Verrechnung von Offsets mit den gepufferten Sollvorgaben korrigiert. Der Korrekturwert der Offsets ist jedoch aufgrund möglicher Kollisionen mit der Merkmalskontur und den benachbarten Bahnen begrenzt. Die realisierten Funktionen der NC-Steuerung müssen daher schnellstmöglich neue, angepasste Bahnen generieren.

Die Reaktionszeit bis zur Bereitstellung neuer Bahnvorgaben wird wesentlich durch die der WZL-NC zugrundeliegende Steuerungsarchitektur und ihre Datenverwaltung beeinflusst. Zusätzlich zu den Rechenzeiten der einzelnen Module, die vom Interpreter über die Werkzeugkorrektur, Transformation und Geschwindigkeitsführung bis zum Interpolator zu durchlaufen sind, kommt es zu Zeitverzögerungen durch die Kommunikation zwischen den Modulen. Insbesondere die Verbindung beiden Hardwareplattformen über das Hallen-Ethernet ist zeitintensiv und verhält sich nicht deterministisch. In der nachfolgenden Tabelle sind die gemessenen Reaktionszeiten für den Austausch von Störungsmeldungen, Positionswerten, Nachrichten und die Zeit zur Neuberechnung oder Adaption der NC-Workingsteps einer Taschenbearbeitung dargestellt, Tabelle 8.2.

	PowerPC (Motorola 333MHz)	PowerPC <- Ethernet -> Industrie-PC	Industrie-PC (Intel Pentium 400MHz)
Adaption oder Neuberechnung der NC-Workingsteps			
Rechtecktasche			< 15 ms
WZL-Logo (Kontur aus 24 Liniensegmenten)			< 15 ms
Komplexe Tasche (100 Linien- & Kreissegmente)			< 15 ms
Reaktion auf das Setzen einer Variablen	(z.B. Lage-Istwert)		
Leerlauf	< 0,2 ms	4 - 25 ms	< 3 ms
Bearbeitung	< 0,2 ms	5 - 30 ms (z.T. bis 100 ms)	< 3 ms
Aufruf eines Prozessobjekts	(z.B. Rückwärtstranformation, Nachrichten)		
Leerlauf	< 0,15 ms	2 - 5 ms	< 3 ms
Bearbeitung	< 0,15 ms	2 - 12 ms (z.T. bis 100 ms)	< 3 ms

Tabelle 8.2: Reaktionszeiten der Kommunikationsmechanismen der WZL-NC

Response times of the WZL-NC measured for different communication methods

Um die Anzahl der Kommunikationsvorgänge zu minimieren, wurden alle zur Adaption auszutauschenden Daten als Nachrichtenobjekte gekapselt und den modulübergreifend aufzurufenden Prozessobjekten als Parameter übergeben. Im Interpreter werden daraufhin die NC-Workingsteps adaptiert und an die nachfolgenden Module der Geometriekette zur Verarbeitung übergeben. In Bild 8.11 ist exemplarisch eine Zeitmessung unter VxWorks für die Adaption einer Taschenbearbeitung dargestellt. Die dabei durchlaufen Funktionen und deren Zeitanteile werden nachfolgend detailliert beschrieben.

Der Eingriff und die Korrekturparameter werden zuerst an den Interpolator gemeldet ❶. Als letztes Softwaremodul vor der Antriebsschnittstelle hat er den schnellsten Zugriff auf die Ist-Positionen und den aktuell ausgeführten Bahn-Workingstep. Der Interpolator wertet den Eingriff aus und bestimmt ausgehend von seinen internen Strategien einen Synchronisationspunkt, bis zu dem er kollisionsfrei Offsetwerte verrechnen kann oder der ohne eine Reduktion der Vorschubwerte in einer steuerungsseitig parametrierten Zeit erreicht werden kann ❷. Dabei nutzt er die von den NC-Workingsteps generierten Zusatzinformationen, die anzeigen, ob eine Bahn adaptiert werden kann, oder ob sie aufgrund der Schnittbedingungen für Eingriffe gesperrt ist. Über ein Prozessobjekt des Transformationsmoduls werden die Achskoordinaten in kartesische, für den Interpreter auswertbare Werkstückkoordinaten zurückgerechnet ❸. Den Synchronisationspunkt meldet der Interpolator zusammen mit den Referenzen des aktuellen NC-Workingstep an den Interpreter ❹. Dieser Schritt ist aufgrund der Ethernetkommunikation zeitintensiv und dauerte in den Versuchen je nach Netzwerkauslastung bis zu 100ms, Tabelle 8.2. Der Interpreter gibt die empfangenen Informationen an seine NC-Workingsteplisten weiter. Die in den Sub-Workingsteps gespeicherten Bahnsegmente erkennen den akti-

ven NC-Workingstep und starten ausgehend von diesem Segment mit der Adaption der nachfolgenden Bahnen. Hierzu übernimmt der die aktuelle Bearbeitungsebene beschreibende NC-Workingstep den Synchronisationspunkt als Startpunkt in die von ihm neu zu berechnende Schnittaufteilung. Die Neuberechnung läuft zeitoptimiert ab, da die ursprünglich bestimmten Bisektoren und Kontursegmente noch im NC-Workingstep gespeichert und mit dem Abarbeitungsfortschritt fortlaufend aktualisiert worden sind. Somit ist die Berechnungszeit weitestgehend unabhängig von der Komplexität der Tasche (vgl. Tabelle 8.2). Das zu zerspanende Restvolumen wird in weniger als 15 ms in neue, in ihrem Abstand korrigierte, asymptotische Bahn-Workingsteps aufgelöst ❺.

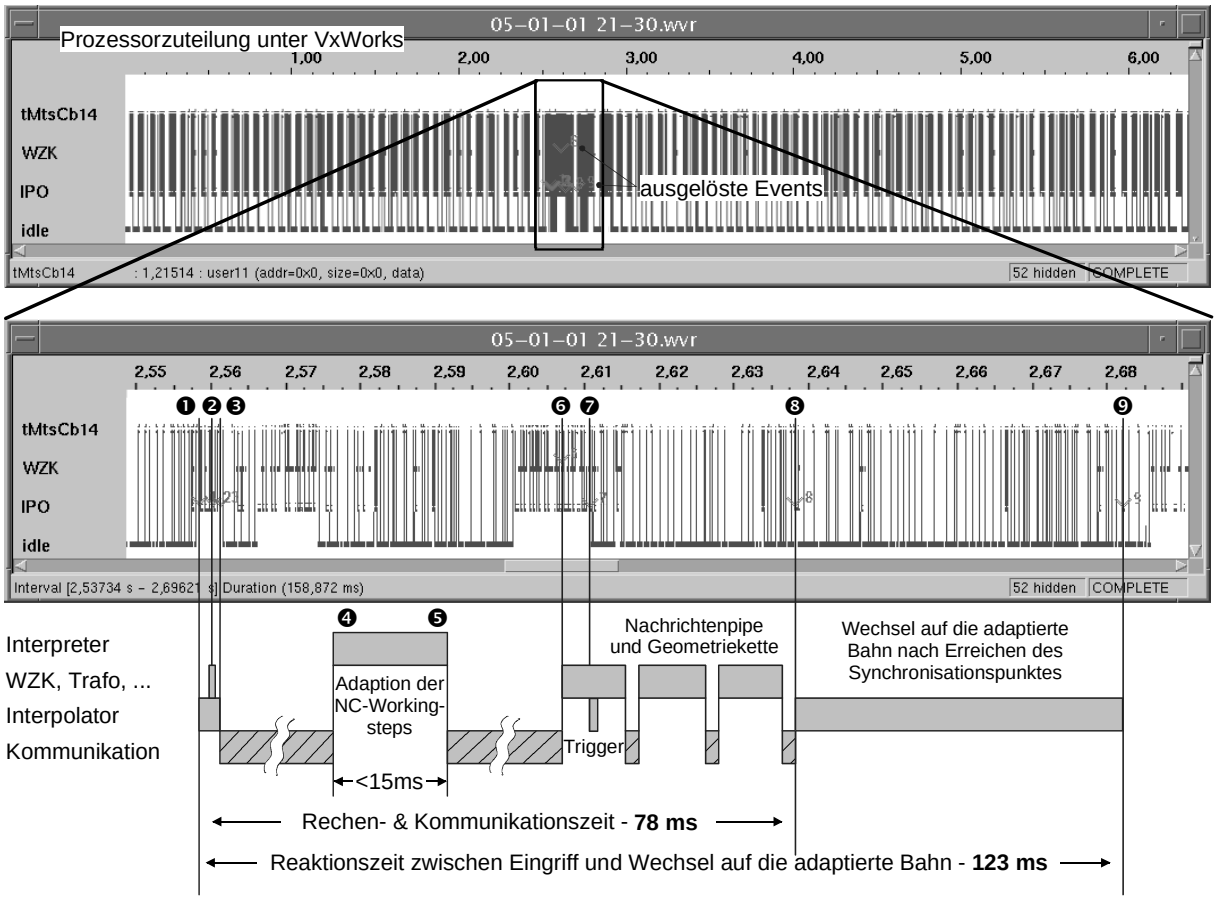


Bild 8.11: Zeitanteil der Kommunikation zwischen den Modulen der WZL-NC
Delta times of the communication between modules of the WZL-NC

Sobald die ersten adaptierten NC-Workingsteps generiert sind, starten diese in parallelen Tasks selbstständig mit ihrer Weiterverarbeitung und werden über das Frontend der WZL-NC als Nachrichten an die nachfolgenden, auf der VxWorks Plattform installierten Module der Geometriekette verschickt. Dort ❻ werden die Bahnsegmente gepuffert, um sie auf die Spline der WZL-NC umzusetzen, zu transformieren und das Geschwindigkeitsprofil zu berechnen. Parallel wird die hoch-priore Nachrichtenpipe der WZL-NC genutzt, um den Module vorab den neuen Startpunkt zu schicken und die Verarbeitung der alten, nicht mehr relevanten Nachrichten abzubrechen ❼. Sobald der erste Spline im Interpolator ankommt ❽, passt der Interpolator die Bahngeschwindigkeit mithilfe seines Interpolationstaktes an die Vorgabe im Synchronisationspunkt an. Ab diesem Punkt wird dann die korrigierte Bahn ruckfrei und ohne eine Programmunterbrechung fortgesetzt ❾.

Käme es aufgrund der Kommunikationsstrecken zu Verzögerungen, würde der Interpolator die Bahngeschwindigkeit über seinen Interpolationstakt reduzieren. Diese Strategie wird ak-

tiv, sobald die hoch-priore Nachricht mit dem ersten adaptierten Segment eine ebenfalls steuerungsseitig parametrierte Grenzzeit überschreitet. Im Worst-Case regelt der Interpolator die Geschwindigkeit auf Null ab und fordert zu dem dann erreichten Punkt eine neue Bahn an.

Alle gemessenen Reaktionszeiten lagen zwischen 80 und 240ms. Hierüber konnte effektiv ein instabiler Prozess mit überhöhten Kräften und starken Schwingungen durch die Reduktion der Werkzeugüberdeckung korrigiert werden. Die Verzögerungszeit war für den eingreifenden Werker praktisch nicht erkennbar.

9. Zusammenfassung und Ausblick

Conclusion and future prospects

An der Werkzeugmaschine wird der Werker heute unzureichend bei der Korrektur und dem Optimieren von NC-Programmen unterstützt. Die einfachen Programmbefehle beschreiben nur den steuerungstechnischen Ablauf der Bearbeitung. Die geplanten, abzuarbeitenden Aufgaben, Merkmale und Operationen sind dagegen für den Werker nicht mehr erkennbar. Ausgehend von dieser reduzierten Information kann er die Bearbeitung nur bedingt nachvollziehen. Die Folge sind Fehler beim Erstellen, Ändern und Einfahren der NC-Programme.

Darüber hinaus stehen dem Werker zur Korrektur der laufenden Bearbeitung nur wenige, auf die Steuerungsparameter oder einzelne Variable beschränkte Eingriffsmöglichkeiten zur Verfügung. Änderungen in der Bahnführung oder an den Werkzeugen erfordern den Abbruch der Bearbeitung, um offline neue Weg- und Schaltbefehle zu ermitteln und diese im NC-Programm zu kodieren. Dadurch verursacht die NC-Programmierung insbesondere in der Prototypen- und Kleinserienfertigung Stillstandzeiten an der Maschine und hohe Kosten. In der Großserie kann dieser Aufwand zwar auf eine größere Stückzahl umgelegt werden, jedoch schränkt die fehlende Durchgängigkeit zwischen den Planungs- und den Programmdaten die Übernahme von Konstruktionsänderungen und Optimierungen in die Variantenplanung ein.

Diese Defizite wurden in der vorliegenden Arbeit analysiert und daraus Anforderungen an die NC-Programmierung, die Unterstützung des Werkers und an die Programmverarbeitung durch die NC-Steuerung abgeleitet. Seitens der Programmierung stehen im Kern dieser Anforderungen die durchgängige, das Prozessverständnis des Anwenders umfassend abbildende Informationsbereitstellung von der Planung bis zur Maschine und die Möglichkeit, Änderungen ohne Konvertierungsverluste direkt in alle Bereiche durchzustellen. Steuerungsseitig resultiert die Forderung, diese Informationen zur Fortschrittsverfolgung anzuzeigen, Überwachungssysteme aufgabenorientiert zu parametrieren und erweiterte, die programmierten Vorgaben einbeziehende Eingriffsmöglichkeiten in die laufende Bearbeitung bereitzustellen.

9.1. Durchgängige, aufgabenorientierte NC-Programmierschnittstelle

Universal, task oriented NC-programming interface

Die geforderte Informationsbereitstellung wurde gelöst, indem ein Datenmodell entwickelt und in der ISO 14649 standardisiert wurde, das sowohl die Planungsergebnisse als auch die steuerungsorientierten Befehle abbildet und aufgabenorientiert miteinander verknüpft. Das Ergebnis ist die vollständige, hierarchisch strukturierte Beschreibung der Bearbeitungsaufgabe. Ausgehend von einem Arbeitsplan und den darin enthaltenen Arbeitsschritten wird sie schrittweise über die zu bearbeitenden Merkmale, Operationen, Strategien, Technologieparameter und Werkzeuge bis hin zu der Vorgabe expliziter Bearbeitungsbahnen und Maschinenfunktionen detailliert. Der Detaillierungsgrad resultiert aus der geforderten Prozessoptimierung sowie den in der Steuerung implementierten Funktionen zur Programmauflösung und Bahngenerierung. Zudem erlaubt er die Migration von konventionellen, befehlsorientierten Steuerungen hin zu autonomen Systemen.

Die entwickelte Lösung erlaubt erstmals den durchgängigen Einsatz des NC-Programms von der Planung bis in die NC-Steuerung. Anwendungsspezifische Zwischenformate, wie APT, CL-DATA oder G-Code, in denen jeweils die höherwertigen, strukturierenden Kontextinformationen der vorangegangenen Anwendungen verloren gehen, sind nicht mehr erforderlich. An der Maschine kann jederzeit auf eine für den Werker verständliche Aufgaben-, Geometrie- und Operationsbeschreibung zurückgegriffen werden. Die aus der Planung durchgängig ver-

fügbaren Informationen ermöglichen zudem eine umfangreiche Unterstützung an der Maschine, so dass Programme schnell und sicher erstellt, simuliert und korrigiert werden können. Die Austauschbarkeit der Programme und die Vorteile für den Anwender wurden mithilfe der vorgestellten Prototypen erfolgreich demonstriert und mit Industriepartnern evaluiert.

Entgegen den vorherigen Ansätzen wurden die Datenmodelle und das Format der NC-Programmierschnittstelle im Rahmen dieser Arbeit vollständig konform zu der STEP Normenreihe ISO 10303 aufgebaut. Bestehende Definitionen zur Abbildung der Konstruktionsgeometrie und der Bearbeitungsmerkmale wurden aus den STEP Applikationsprotokollen AP 209, 214 und 224 übernommen. Das war die Grundlage, um die hier entwickelte NC-Programmierschnittstelle in industrienahen Prototypen implementieren, international abgleichen und erfolgreich als ISO 14649 standardisieren zu können. Die allgemeingültigen Elemente zur Beschreibung des Bearbeitungsablaufs sowie der Spannlagen wurden in einem Grundmodell zusammengefasst (ISO 14649 – Part 10) und technologiespezifisch um Partialmodelle für das Fräsen (Part 11), Drehen (Part 12), Erodieren (Part 13, 14) sowie das Messen und Prüfen ergänzt. Die Leistungsfähigkeit des hier entwickelten Fräsmodells ist so umfassend, dass es als Applikationsprotokoll in den STEP Standard übernommen wurde [ISO10303 -238] und durch die OMAC Gruppe eine industrielle Umsetzung verfolgt wird [KENN07].

Die entwickelte NC-Programmierschnittstelle umfasst bisher nur Teilbereiche der durch NC-Programme anzusteuernden Bearbeitungsprozesse und Automatisierungssysteme. Die industrielle Akzeptanz und Umsetzung wird wesentlich von der Entwicklung weiterer Partialmodelle abhängen. Zudem müssen Institute und Industriearbeitskreise fortlaufend den Funktionsumfang der Planungssysteme und NC-Steuerungen abgleichen, um die NC-Programmierschnittstelle rechtzeitig zu ergänzen, ehe dass sich analog zu den NC-Zyklen herstellerspezifische Varianten technisch vergleichbarer Operationen und Merkmale etablieren.

In einem nächsten Schritt müssten unter anderem Modelle zur Beschreibung der Maschinen, Spannmittel und Werkzeuge erstellt werden. Diese Daten sind in der vorliegenden Arbeit, deren Fokus auf den Featuren und Operationen lag, nur grob modelliert und nur zum Teil standardisiert worden. Diese Informationen sind eine weitere Grundlage, um Prozesse in der Steuerung autonom ohne Eingriffe des Werkers optimieren zu können.

9.2. Autonome, adaptierbare Programmverarbeitung in der NC-Steuerung

Autonomous, adaptable processing of the NC-program in the numeric control

Die im Rahmen dieser Arbeit entwickelte NC-Steuerung übernimmt die umfangreichen Informationen der NC-Programmierschnittstelle und führt sie mithilfe ihrer erweiterten, internen Datenstruktur durchgängig bis zu ihren I/O- und Antriebsschnittstellen fort. Hierzu wurden interaktiv verknüpfte Softwareobjekte, die NC-Workingsteps eingeführt. Die Attribute der NC-Workingsteps speichern die Programmvorgaben, die von den Steuerungsmodulen oder den Objektmethoden generierten Zwischenergebnisse und die final auszuführenden Sollwerte. Um die Komplexität der einzelnen NC-Workingsteps zu begrenzen, wurde eine Bibliothek aus spezialisierten NC-Workingsteps entwickelt. Diese bauen systematisch aufeinander auf. Zur Programmumsetzung lösen sie sich ausgehend von übergreifenden Arbeitsschritten und Operationen schrittweise in Listen aus Sub-Workingsteps auf, bis sie in ihrer elementarsten Form explizite Maschinenfunktionen oder Sollwerte kapseln. Da die NC-Workingsteps als eine offen erweiterbare Bibliothek in die Steuerung integriert sind, kann sie ein dritter Entwickler um eigene Abläufe und Bahnalgorithmen ergänzen.

Über die NC-Workingsteplisten bleibt der Bezug zu den programmierten Aufgaben, Merkmalen und Operationen durchgängig erhalten. Auf dieser Grundlage ist die Fortschrittsverfolgung nicht mehr auf die Anzeige einfacher Steuerungsbefehle sowie die der Soll- und Istwerte beschränkt. Der Werker kann die Abarbeitung grafisch animiert verfolgen und hat jederzeit

den Bezug zu den übergeordneten Planungsvorgaben. Darüber hinaus liefern die hochwertigen Geometriedaten des Rohteils und der Bearbeitungsmerkmale die Grundlage, um Kollisionen zu erkennen und autonom in der Steuerung sichere Positionier, Anrück-, Bearbeitungs- und Freifahrbewegungen zu generieren. Diese Funktionen unterstützen den Werker effektiv und befreien ihn von der abstrakten sowie fehleranfälligen Bestimmung einzelner Wegsegmente und Technologiebefehle.

Ferner gleichen sich die entwickelten NC-Workingsteps selbstständig über ihre Methoden untereinander ab. Dieses Konzept bildet den Kern, um in einer laufenden Bearbeitung die NC-Programmvorgaben ändern und so umfassend in den Prozess eingreifen zu können. Die übergreifenden, einzelne Arbeitsschritte und Operationen abbildenden NC-Workingsteps werten die externen Eingriffe anhand ihrer spezifischen Prozessmodelle aus, melden Fehler an den Benutzer und adaptieren selbstständig ihre Vorgaben sowie die der untergeordneten und nachfolgenden NC-Workingsteps. Indem diese Funktionalitäten aufeinander aufbauen, stehen sie unmittelbar in erweiterten NC-Workingsteps zur Verfügung und unterstützen die Entwicklung neuer Prozessabläufe und Eingriffsmethoden. Da die Umsetzung des Eingriffs in den NC-Workingsteps realisiert wird, lassen sich eigene Entwicklungen einfach als Bibliotheken in der Steuerung ergänzen und die internen Algorithmen, d.h. das Prozesswissen schützen.

Um die Reaktionszeiten nach einem Eingriff zu minimieren, generieren die NC-Workingsteps präventiv Übergangs- und Abbruchstrategien. Bezogen auf die Bewegungsführung wurde dazu die entwickelte NC-Steuerung um Offsetvektoren erweitert, die über die Achsbewegungen parametriert werden. Auf diese Vektoren werden die im NC-Programm kodierten, den Bahnverlauf bestimmenden Prozessparameter übertragen und in der Geometriekette bewegungssynchron auf die Maschinenkinematik abgebildet. Im Eingriffsfall werden die Offsetwerte kontinuierlich und richtungsorientiert mit den geplanten Sollwerten verrechnet, bis die adaptierte Bahn berechnet, weiterverarbeitet und zur Ausführung bereitgestellt wird.

Die entwickelten Vorteile einer aufgabenorientierten Prozessverfolgung und der steuerungsinternen Merkmal- und Operationsauflösung lassen sich in bestehenden NC-Steuerungen mit geringem Aufwand realisieren. Dazu werden die vorhandenen Algorithmen zur Abarbeitung der herstellerspezifischen NC-Zyklen auf eine standardisierte Programmierschnittstelle, wie die hier erarbeitete ISO 14649, umgestellt. Auf dieser Grundlage arbeiten dann die Planungs- und Programmiersysteme sowie die NC-Steuerung durchgängig mit einem Format ohne Informationsreduzierung. Die Umstellung wird dabei einerseits von der Weiterentwicklung der ISO 14649 abhängen und zudem von den CAP/CAM Systemen, die letztlich die Programme generieren müssen. Wichtig hierbei ist, dass sowohl in den CAM-Systemen als auch in den NC-Steuerungen standardisierte Softwareschnittstellen geschaffen werden, um die Algorithmen durchgängig von der Planung bis in die Abarbeitung einsetzen zu können, gleichsam jedoch die darin gekapselte Kompetenz der Hersteller zu schützen.

Weiteres Potential liegt in der Prozessrückführung durch die NC-Workingsteps. Sie nehmen an den Sollwertschnittstellen den realen Prozessverlauf auf und ordnen diesen eindeutig einzelnen Operationen und Aufgaben zu. In der Planung könnten diese Informationen zur Bewertung der gespeicherten Operationen und Technologiedaten ausgewertet werden. Würden die aktuell in die CAM-Systeme integrierten virtuelle NC-Kerne um diese NC-Workingstep erweitert, so könnten sie die real aufgezeichneten internen Steuerungsabläufe als Störung in die Simulation einfließen lassen.

Conclusion and future prospects

Today, the operator is insufficiently supported at the machine tool while correcting or optimizing machining programs. These programs only describe a sequence of simple toolpath and switching commands to be executed by a numerical control. They do not provide any planning information, like working steps, features or operations to the operator. Based on this poor level of information he can hardly review the machining process. The consequences are errors while generating, editing or optimizing machining programs in the shop-floor.

At runtime the machining program cannot be changed. There are only few parameters and variables which can be used for process manipulation at runtime. A programmed toolpath or a selected tool can only be changed after stopping program execution. This causes downtimes at the machine tool and high costs. Even if these downtimes can be compensated in mass production by a larger quantities of parts, the lack of consistency between planning data and machining program limits the feedback of changes and optimizations at the NC.

In a first step this thesis analyzed these deficits and defined the requirements of NC programming, operator support and manipulatable program processing. On the part of programming these requirements focus a solution, providing consistent and universal information from planning to machining that reflects the process understanding of the operator. Inside the numerical control this information shall be used to display the machining progress and geometry, to parameterize monitoring and control systems according to the actual operation and to offer enhanced manipulation routines in order to optimize the process at runtime.

Universal, task oriented NC-programming interface

The information retrieval was solved by developing and standardizing a data model that contains both complete planning data and control oriented commands. The result is a comprehensive, hierarchically structured description of machining jobs. The jobs are defined in so called *Workingsteps* which are listed in *Workplans*. Each *Workingstep* consists of a geometry to be machined (e.g. a *Feature*) and an *Operation* specifying which *Strategy*, *Technology* parameters and *Tool* shall be used. In addition explicit *Toolpaths* and *NC-Functions* can be defined within the *Operation* in order to precisely predetermine how to machine the part and when to activate or deactivate the machine's functions. The level of detail depends on the requested degree of optimization and the availability of functions to compile toolpaths and commands autonomously inside the numerical control. In addition it permits to migrate step by step from a conventional numerical control and its detailed commands to a more flexible system that can process features based on the given operations, tools and machine data.

The developed solution permits for the first time the universal use of NC programs from planning to numerically controlled machining. Specific formats, like APT, CL-DATA or G-codes which convert and simplify the planning information are not necessary any longer. At the machine tool the operator can access at any time an understandable description of the machining task, geometry and operation. The information constantly provided from planning allows manipulating even complex NC programs at the machine tool. Hence NC programs can be corrected or optimized more easily and fast with less risk of wrong codes or misunderstandings. The exchangeability of the programs and the advantages for the user were successfully demonstrated based on the realized prototypes and evaluated with industrial partners.

Unlike previous concepts the NC programming interface developed within this thesis is completely conform to the STEP standards (ISO 10303). Existing definitions to define general geometry or machining features were taken over from the STEP application protocols AP 209, 214 and 225. By consequence the model was internationally accepted, implemented in industrial prototypes and finally successfully standardized in the ISO standard ISO 14649. The general elements of the data model, like the workplan or the set-up were grouped in a general model (ISO 14649 – Part 10). The technology specific models were described in spe-

cialized models for milling (Part 11), turning (Part 12), EDM (Part 13, 14) and drafted for inspection. Especially the model for milling, which has been a main task of this thesis, could be completed to an extend that it was transferred to a STEP application protocol [ISO10303-238] and the OMAC group started applying it in industry [KENN07].Up to now the NC programming interface was only defined for a couple of machining technologies. The industrial acceptance will substantially depend on the development of further models. In addition institutes and industrial working groups need to review the models constantly, in order to add new elements to the interface that can handle the future process and geometry information of CAM, CAP and NC.

In addition data models for machine tools, clamping devices and cutting tools need to be developed. This thesis only drafted some of the required elements as it focused the geometrical and operational data. In order to autonomously optimize the machining process inside the NC-control without the help of any operator this information is mandatory.

Autonomous, adaptable processing of NC-programs in the numeric control

After defining and standardizing the new NC programming interface ISO 14649 a numerical control was developed to process this extensive information. The realized numerical control is able to process the operations and features of the NC program at runtime. In order to constantly keep the relations between workingsteps and generated set-values so called NC-Workingsteps have been introduced. In a first step these NC-Workingsteps store the input received from the NC program. Then they process the information and generate toolpaths, sequences of switching commands and set-values to feed the drives. In order to handle the complexity of this task a NC-Workingstep library was developed containing numerous subtypes of NC-Workingsteps which are specialized to process different NC program data. They systematically complement one another. Processing the content of the NC program the NC-Workingsteps split themselves up into Sub-Workingsteps. The final level of Sub-Workingsteps contains simple NC-Workingsteps that handle explicit machine functions or set-values only. Since the NC-Workingsteps are integrated into the numerical control as an exchangeable library an external developer can easily add his own NC-Workingsteps to realize special toolpath algorithms etc.

The invented NC-Workingsteps are structurized in so called Sub-Workingstep lists. These lists preserve the context to the workplan, workingsteps, features, operations and tools defined in the NC program. This solution allows to pursue the machining progress based on understandable, high level process information or features instead of listing lines of G- and M-codes. The operator can view a graphically animated machine tool with moving axes and tools removing material from a simulated work-piece. He always can refer to the original workingsteps, operations and features. In addition the high level geometry data of the raw part, the finished part and the tools can be used to detect collisions and to autonomously generate optimized toolpaths, positioning or retract movement. These functions were realized in different prototypes meeting the previously defined requirements. The solution proved to support the operator at the machine tool while generating or editing NC programs, observing a running operation or optimizing the process at runtime. The solution substituted the error-prone coding with abstract G-codes and provided new, additional functionality.

The future task will be to support industry to migrate from today's G-Codes to ISO 14649. Actually CAP/CAM vendors are waiting for NCs to process ISO 14649 while the NC vendors are waiting for planning systems that are able to generate ISO 14649 program files. A first migration step could be to closer link CAM systems and the NC. A so called CAM-Client could directly run on the NC's HMI. Thus the operator could use CAM dialogues and views to generate or edit NC programs. The simulation routines of the CAM system could be used to detect collisions before executing a toolpath segment and to display the running machining process. In a next step either the numerical control has to provide fast access to its kernel or

the CAM functionality to re-generate alternative toolpaths and workflows has to be integrated into the NC. Both solutions will enhance the functionality at the NC and inside the NC.

10. Literaturverzeichnis

References

ALBE00 Albert, M.: STEP-NC – The End of G-Codes. In: Modern Machine Shop. Gardner Publications, 2000, S. 70-80

APZ02 Sonderforschungsbereich 368 „Autonome Produktionszellen“. Arbeits- und Ergebnisbericht 2000-2002, RWTH Aachen, 2002

ARNO46 Arnold, R. N.: Mechanism of Tool Vibration in Cutting of Steel; Proc. I. Mech. E. 154; 1946

ARTI00 Computer integrated Tool and Machine Monitoring (CTM). In: Firmenprospekt der ARTIS Gesellschaft für angewandte Messtechnik mbH, 2000

ARTI02 Prozessüberwachung und Spindelüberwachung an einem Bearbeitungszentrum. In: Anwendungsbericht der ARTIS Gesellschaft für angewandte Messtechnik mbH, 2002

ARTI03 N.N.: Adaptive Control – Grenzwertregelung zur Optimierung der Standzeit und Bearbeitungszeit durch Vorschubanpassung. In: Firmenprospekt der ARTIS Gesellschaft für angewandte Messtechnik mbH, 2003

AWF85 AWF – Ausschuss für Wirtschaftliche Fertigung e.V. (Hrsg.): Integrierter EDV-Einsatz in der Produktion, CIM Computer Integrated Manufacturing. Eschborn, 1985

BALZ01 Balzert, H.: Lehrbuch der Softwaretechnik – Software-Entwicklung. Spektrum Akademischer Verlag, Heidelberg, 2001. Bd. 1

BAPS97 Dr.-Ing. K. Brankamp System Prozessautomation GmbH: Schnittstellenbeschreibung BAPSI (Brankamp – Artis – Prometec – Standard – Interface). 1997. Firmenschrift.

BEER94 Beer, C.: CAD/CAP unterstützte Generierung ratterfreier NC-Programme für das Schaftfräsen. Dissertation RWTH Aachen. Aachen, 1994

BENK95 Benkler, H.: Grundlagen der NC-Programmiertechnik für Ausbildung, Arbeitsplanung und Fertigungspraxis. Carl Hanser Verlag, München, 1995

BEUK99 Beuke, D.; Conrad, K.-J.: CNC-Technik und Qualitätsprüfung. Hanser, München Wien, 1999

BITZ96 Bitzel, H.: Faszination Blech. Flexible Bearbeitung eines vielseitigen Werkstoffs. Dr. Josef Raabe Verlags GmbH, Stuttgart Berlin, 1996

BOSC96 Robert Bosch GmbH: Typ 3 osa. Dokumentation der NCS. Erbach, 1996. Firmenschrift

BOSC01 Robert Bosch GmbH: Typ3 osa. Dokumentation der Kundenschnittstelle Bewegungserzeugung. Erbach, 2001. Firmenschrift.

BOSC04 Bosch Rexroth AG: Rexroth PNC – Offenes Steuerungs- und Antriebssystem für CNC-Werkzeugmaschinen. Lohr, 2004. Firmenschrift

BOSC04a Bosch Rexroth AG: IndraMotion MTX – Bedienungsanleitung. Lohr, 2004. Firmenschrift

BOSC04b Bosch Rexroth AG: Rexroth IndraMotion MTX – The new CNC system solution – open, complete, high performance. Lohr, 2004. Firmenschrift

BOLL04 Bollig, A.: Prädiktive Prozessregelung beim Schweißen mit Laserstrahlung. Dissertation, RWTH Aachen. VDI Verlag GmbH, Düsseldorf, 2004

BREC04 Brecher, C.; Wolf, J.: Prozessabhängige Bahnplanung beim Taschenfräsen. Weniger Überlasten und Rattereffekte mit einer Bahnplanung zur Laufzeit. In: wt – Werkstattechnik online. 2004, Nr. 7/8, S. 350-354

BREC06 Brecher, C.; Vitr, M.; Wolf, J.: Closed-loop CAPP/CAM/CNC process chain based on STEP and STEP-NC inspection tasks. In: Int. Journal of Computer Integrated Manufacturing, Vol. 19, September 2006, S. 570-580

BREU83 Breuer, Friedel: Adaptive Control und automatische Schnittaufteilung – Steuerungsspezifische Zusatzsysteme zur Produktivitätssteigerung beim NC-Drehen. Dissertation RWTH Aachen, 1983

BRÜH97 Brühl, J.: OSACA und HÜMNOS technische Ergebnisse – Trendwende in der Steuerungstechnik, EMO Hannover, 1997

BRÜH03 Brühl, J.: Methoden zur Realisierung einheitlicher Schnittstellen für offene Steuerungssysteme. Dissertation RWTH Aachen, 2003

BROU00 Brouër, N.; NC-Steuerungskern mit Datenschnittstelle für eine Autonome Produktionszelle. Dissertation RWTH Aachen, 2000

BUDD70 Budde, W.: Arbeitsablauf und Werkzeugermittlung für Drehbearbeitung – Ein Beitrag zur Automatisierung der Fertigungsplanung. Dissertation RWTH Aachen, 1970

CADC02 N.N.: Ein Rückblick auf die CAD/CAM-Geschichte. In: CAD-CAM Report. 2002, Nr. 6, S. 46-63

CADC04 Durchgängiger Datenfluss bis in die Fertigung. Marktübersicht CAM-Systeme. In: CAD-CAM Report Nr.7 (Teil I) und 8. (Teil II), 2004

CASC07 Bibliothek und Dokumentation zur 3D Grafiklibrary Open Cascade. URL: www.opencascade.org [Stand: 2007]

CGTE04 CGTech GmbH: Vericut 5.4 – neue und verbesserte Funktionen. 5/2004. Firmenschrift

DANI96 Daniel, C., Dynamisches Konfigurieren von Steuerungssoftware für offene Systeme. Dissertation, Universität Stuttgart, 1996

DAUD01 Daude, R.: Mobiles Maschinen- und Prozessinteraktionsystem. Dissertation RWTH Aachen, 2001

DAXL04 Daxl, J.; Kurz, S.; Schachinger, W.: Grundlagen über numerisch gesteuerte Werkzeugmaschinen (CNC). Programmierung nach ISO für Sinumerik 840D. Verlag Jungend & Volk, Wien, 2004

DIN89 Deutsches Institut für Normung DIN: Die Schnittstellen der rechner-integrierten Produktion (CIM) – CAD und NC-Verfahrenskette. DIN-Fach-bericht 20, Beuth-Verlag, 1989

DIN66025 DIN 66025: Programmaufbau für numerisch gesteuerte Arbeitsmaschinen. Berlin: Beuth-Verlag, 1983 (Teil 1), 1988 (Teil 2).

DEIT04 Deiter, F.: Die Automation der Kleinserie ist spannend. In: Werkstatt und Betrieb.06/2004, S. 149-151

DEUS98 Deuse, J.: Fertigungsfamilienbildung mit feature-basierten Produktmodell-daten. Dissertation. RWTH Aachen, 1998

DENK02 Denkena, B.; Tönshoff, H. K.; Selle, J.; Storr, A.; Heusinger, S.; Rogers, G.: Offline-Berechnung der Zerspankräfte in der NC-Programmierung – Vorhersage der Zerspankräfte beim HSC-Schlichtfräsen. In: wt – Werkstattstechnik online. 2002, Nr. 11/12, S. 581-587

DOMB93 Dombrowski, M.; Müller, H.D.; Petereit, J.: Qualifizierung als Werkstatt-Orientierte-Produktionsunterstützung. In: CIM-Management, Nr. 3, 1993, S. 45-51

DOYL04 Doyle, P.: Introduction to Real-Time Ethernet II. In: The Extension – A Technical Suppelment to Control Network. Vol.5, Issue 4, 2004 (www.ctrlink.com/pdf/volume5n4.pdf)

DOWE03 Dowe, M.; Eberle, C.; Stalla, B.: Werkzeugpotentiale optimal nutzen, Werkstatt und Betrieb 9/2003, Carl Hanser Verlag, München, 2003

ENGS02 Engstler, M.: NC-Prozesskette in Kleinbetrieben. Anforderungen, Verfahrensauswahl, Systemeinführung und Erweiterungen. Dissertation. Universität der Bundeswehr München, 2002

EMDE07 Emde, C.; Gleixner, T.: Standard-Linux wird echtzeitfähig. In: Elektronik. 2007, Nr. 3, S. 62-69

EVER94 Eversheim, W. Et al.: NC-Verfahrenskette. Hrsg.: H.J. Warnecke, R. Schuster, DIN. 1. Aufl. Berlin: Beuth, 1994

EVER97 Eversheim, W.: Organisation in der Produktionstechnik 3 – Arbeitsvorbereitung. 3. Auflage. Berlin: Springer, 1997. (Band 3)

EVER01 Eversheim, W.; Phornprapha, M.; Westekemper, M.: Migrationskonzept für DV-Systeme der NC-Verfahrenskette. Einbindung feature-basierter Planungsmethoden in technologieorientierte NC-Programmiersysteme. In wt Werkstatttechnik. 03/2001, S. 117

EVER02 Eversheim, W.: Organisation in der Produktionstechnik. Band 3. Arbeitsvorbereitung. 4. Aufl. Springer, Berlin, 2002

EXAP07 EXAPT: Durchgängiger bidirektionaler Informationsfluss zwischen NC-Programmiersystem, NC-Steuerung und Antrieb durch Feature-basierten Datenaustausch. [Stand September 2007]
URL: www.exapt.de/de/fvp/Durchgaengiger-Informationsfluss.htm

FANU04 GE Fanuc Automation: Handbuch Bedienung. 2004. Firmenschrift

FANU04a Machine Tool Productivity with i-Adapt. In: Produktkatalog zu i-Adapt von GE Fanuc Automation. 2004. Firmenschrift

FAUS97 Fauser, M.: Steuerungstechnische Maßnahmen für die Hochgeschwindigkeitsbearbeitung. Dissertation RWTH Aachen, 1997

FRAN02 Francis, B. R.: What is Post-Processing? www.icam.com [Stand Januar 2002]

FRAN06 Franke, R.: Immer im optimalen Bereich – Lernfähige Software passt permanent den Vorschub an die Bedingungen beim Zerspanen an. In: MM Maschinenmarkt, Nr. 6, 2006, S. 28-29

FRIE00 Fries, E.; Uhlmann, E.: Anwendung neuronaler Netze zur Werkzeugverschleißerkennung beim Fräsen. In: Zukunftsweisende Anwendungen integrierter Sensorsysteme. (Hrsg.). VDI-Fortschrittsberichte Reihe 8 Nr. 846, VDI, Düsseldorf, 2000

GATH77 Gather, M.: Adaptive Grenzregelung für das Stirnfräsen – Leistungsregelung, Ratterbeseitigung, automatische Schnittaufteilung. Dissertation RWTH Aachen, 1977

GEHR89 Gehrke, U.: Anforderungen an fertigungsorientierte Schnittstellen für die NC-Programmierung bei Einzelteilfertigung. In: Rechnerunterstützung in der Konstruktion. Düsseldorf: VDI-Verlag, 1989, VDI Bericht 752, S. 191/205

GIES02 Giesa, H.G.; Timpe, K.-P.: Technisches Versagen und menschliche Zuverlässigkeit: Bewertung der Zuverlässigkeit in Mensch-Maschine-Systemen. In: Timpe, K.-P. u.a. (Hrsg.): Mensch-Maschine-Systemtechnik. 2. Aufl. Düsseldorf: Symposion, 2002, ISBN 3-933814-83-9

GIES73 Gieseke, E.: Adaptive Grenzregelung mit selbstständiger Schnittaufteilung für die Drehbearbeitung. Dissertation, Aachen, 1973

GLOC99 Glockner, D.: Integration von Facharbeiter Erfahrungswissen auf der Basis von Fertigungsfeatures. Dissertation, TU Darmstadt. München: Carl Hanser Verlag, 1999

GOET04 Göttsch, N.: CAM-Integration: Wunsch und Wirklichkeit. In: CAD-CAM Report Nr.2, 2004, S. 26-29

HAAS00 Haasis, D.; Frank, D; Rommel, B.; Weyrich, M.: Features – Rückrat zur Integration der Produkt-, Prozess- und Ressourcengestaltung. In: Konstruktion Zeitschrift für Produktentwicklung, 2000, Nr. 4

HEID04 Heidenhain: Programmierhandbuch iTNC 530. 2004. Firmenschrift

HARD02 Hardebusch, C.: Entwicklung offener Steuerungssysteme. Methoden und Werkzeuge. Diss. RWTH Aachen, 2002

HARR73 Harrington, J.: Computer Integrated Manufacturing. New York: Pub Industrial Press, 1973

HEUS03 Heusinger, S.; Müller, P.: STEP-NC – New Data Interface for NC-Programming. Data Model for Turning. In: STEP-NC Newsletter No. 5, www.step-nc.org [Stand September 2003]

HEUS05 Heusinger, S.: STEP-NC basierter Korrekturkreis für die Schlichtbearbeitung von Freiformflächen. Dissertation Universität Stuttgart. Jost-Jetter Verlag, Stuttgart, 2005

HELD91 Held, M.: On the Computational Geometry of Pocket Machining. Lecture Notes in Computer Science, Bd. 500. Berlin, Heidelberg: Springer, 1991

HENN97 Henning, K. R.: Qualitätsmanagement in der NC-Informationskette – Systematische Informationsrückführung und Aufbereitung von Informationen zur Fehlerprävention. Dissertation Universität Hannover. Düsseldorf: VDI Verlag, 1997

HIRS93 Hirschen-Kreisen, H.: NC-Entwicklung als gesellschaftlicher Prozess. Frankfurt/Main; New York: Campus Verlag, 1993

HOFF05 Hoffmann, M.; Eickenberg, S.; Hack, O.: CAD/CAM mit CATIA V5 – NC-Programmierung, Postprocessing, Simulation. München: Hanser Verlag, 2005

HUEM98 N.N.: Referenzarchitektur der HÜMNOS-Basissteuerung. In: bmb+f Projektträger PFT (Hrsg.). 2. Auflage, Karlsruhe, 1998

INDR96 Indramat GmbH: NC-Compiler V17. Lohr a.M., 1996. Firmenschrift

INDR97 Indramat GmbH: NC-Programmieranleitung V17. Lohr a.M., 1997. Firmenschrift

INDR98 Indramat GmbH: MTC200/MT-CNC. Bohrer-Bruchüberwachung 18VRS. Lohr a.M., 1998. Firmenschrift

ISO10303-11 ISO 10303-11 (IS): Industrial automation systems and integration – Product data representation and exchange – Part 11: Description methods: The EXPRESS language reference manual. Genf: International Organization for Standardisation, 2004

ISO10303-21 ISO 10303-21 (IS): Industrial automation systems and integration – Product data representation and exchange – Part 21: Implementation methods: Clear text encoding of the exchange structure. Genf: International Organization for Standardisation, 2002

ISO10303-224 ISO 10303 AP 224 (DIS): Industrial automation systems and integration – Product data representation and exchange – Part 224: Mechanical Product Definition for Process Planning Using Machining Features. Genf: International Organization for Standardisation, 1997

ISO10303-238 ISO 10303 AP 238: Industrial automation systems and integration – Product data representation and exchange – Part 238: Application Protocol: Application interpreted model for computerized numeric controllers. Genf: International Organization for Standardisation, 2007

ISO14649-1 ISO 14649 Part 1 (IS): Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 1: Overview and fundamental principles. Genf: International Organization for Standardization, 2003

ISO14649-10 ISO 14649 Part 10 (IS): Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 10: General process data. Genf: International Organisation for Standardization, 2004

ISO14649-11 ISO 14649 Part 11: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 11: Process data for milling. Genf: International Organisation for Standardization, 2004

ISO14649-12 ISO 14649 Part 12: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 12: Process data for turning. Genf: International Organisation for Standardization, 2005

ISO6983 ISO 6983: Numerical control of machines – Program format and definition of address words – Part 1: Data format for positioning line motion and contouring control systems. Genf: International Organisation for Standardization, 1982 (First Edition), 1992 (Second Edition)

KAEV04 Kaever, M.: Steuerungsintegrierte Fertigungsprozessüberwachung bei spanender Bearbeitung. Dissertation RWTH Aachen, 2004

KAHM05 Kahmen, A.: Eine offene Integrationsplattform zur Prozess- und Maschinenzustandsüberwachung in Numerischen Steuerungen. Dissertation RWTH Aachen. Aachen: Shaker Verlag, 2005

KENN07 Kennedy, B.: All together now – STEP-NC is a data-rich language designed to overcome the current standard for defining the movement of machine tool axes. In: Cutting Tool Engineering Magazine, No. 7, Vol. 59, 2007

KIEF05 Kief, H.B.: NC/CNC Handbuch 2005/2006. Hanser Verlag, München, 2005

KIENZ57 Kienzle, O.; Victor, H.: Spezifische Schnittkräfte bei der Metallbearbeitung. In: Werkstofftechnik und Maschinenbau 47, 1957, Nr. H5, S. 224-225

KINZL06 Kinzler, E.: Maschinensimulation „Virtuelle Maschine“ von Index – Simulation in einer neuen Dimension. In: motion world, Hrsg. Siemens, 2/2006, S. 10-11

KLEI95 Klein, F.: NC-Steuerung für die 5-achsige Fräsbearbeitung auf der Basis von Nurbs. Dissertation RWTH Aachen, 1995

KLEM03 Klemm, P.; Heusinger, S.: Planungsfehler werden vermieden. STEP-NC basiertes Programmieren der Komplettbearbeitung auf Dreh-Fräszentren verkürzt Programmierzeiten. In: Maschinenmarkt MM das Industriemagazin. 2003, Nr. 34, S. 26-29

KLEM05 Klemm, P.; Selig, A.; Wosnik, M.; Kramer, C.: Rückführung von Prozessdaten mittels STEP-NC. In: wt – Werkstattstechnik online. 2005, Nr. 5, S. 368-372

KLIN04 Klingauf, W.: Unter adaptiver Kontrolle immer am Limit. In: Werkstatt und Betrieb. 06/2004, S. 63-65

KLOC98 Klocke, F.: Produktion 2000 plus: Visionen und Forschungsfelder für die Produktion in Deutschland. Band 1 und 2, WZL, RWTH Aachen, 1998

KLOC04 Klocke, F.; Kratz, H.-H.: Prozessüberwachung. In: Autonome Produktion. Klocke, F.; Pritschow, G. (Hrsg.). Springer Verlag, Berlin, 2004

KOCH96 Koch, M.: Autonome Fertigungszellen – Gestaltung, Steuerung und integrierte Störungsbehandlung. Dissertation TU München, 1996

KOCH05 Koch, J.: Wissensbasiertes CAM-System steigert Effizienz beim Programmieren. In: Maschinenmarkt MM das Industriemagazin, 17.11.2005

LAUS01 Lauscher, J.: Effiziente Methoden für die NC-Programmierung. In: RPD Report NCG, 2001, Nr. 15, S. 18

LANG02 Lange, D.: Hohe Produktivität durch Prozessüberwachung. In: Tagungsunterlagen zum 10. Internationalen Feinbearbeitungskolloquium (FBK) 2002. Braunschweig, 7.-9. Oktober 2002

LANG05 Lange, D.: Alles unter Kontrolle. Prozessüberwachung sichert Qualität beim Fertigen komplexer Teile auf Bearbeitungszentren. In: Maschinenmarkt MM das Industriemagazin, Nr. 15, 2005

LAYG05 Lay, G.; Maloca S.: Aufgabenintegration – Abkehr vom Taylorismus. In: Mitteilungen aus der Produktionsinnovationserhebung. Fraunhofer ISI (Hrsg.), Juli 2005, Nr. 36

MAIE74 Maier, K.: Grenzregelung an Werkzeugmaschinen – Beitrag zur Auslegung und Bewertung von ACC-Systemem. Dissertation, Universität Stuttgart, 1974

MARC93 Marczinski, G.: Verteilte Modellierung von NC-Planungsdaten – Entwicklung eines Datenmodells für die NC-Verfahrenskette auf Basis von STEP. Dissertation RWTH Aachen, Aachen: Shaker Verlag, 1993

MATR97 ESPRIT project 6245 MATRAS – Manufacturing Technology for Complex Geometries Based on Rational Splines. Final Report. Brüssel: Europäische Kommission, 1997

MAV05 N.N.: NC-Simulation als Versicherungspolice. In: MAV – Maschinen Anlagen Verfahren, Nr. 5 / 2005, S. 58 ff.

MENZ03 Menzel, T.; Dolansky, S.; Müller, P.: Feature-orientiertes NC-Sprachelement in Ergänzung der DIN 66025 / ISO 6983. Arbeitspapier des ISO TC 184 / SC1 / WG7. Siemens Erlangen, 2003

MENZ05 Menzel, T.: Mehrwert Simulation – Innovative Systeme und Dienstleistungen steigern Mehrwert für Maschinenhersteller und -anwender. In: Motion World, September 2005, S. 16-17

META06 6D Nahtführung. In: Firmenbroschüre. Meta-Scout GmbH, München, 2006

MICH00 Michaloski, J.; Brila, S.; Yen, J.: Software Models for Standardizing the Human-Machine Interface Connection to a Machine Controller. In. Proceedings of the 2000 World Automation Conference. Juni 2000

MUEN05 Münnich, R. I.: Maschinennahe Qualitätsregelkreise für das Einfahren und Optimieren von NC-Zerspanprozessen. Dissertation RWTH Aachen, 2005

NACS01 Nacsa, J.: Comparison of Three Different Open Architecture Controllers. In: Proc. Of IFAC MIM, Prag, 2.-4. August 2001, S. 134-138

NCFE03 N.N.: Werkzeugüberwachung sichert Schwerzerspanung. In: NC-Fertigung, Mai 3/2003

NCFE05 N.N.: Integriertes Crash Verhinderli – Heidenhain entwickelt dynamische Kollisions-Überwachung. In: NC-Fertigung, Juni 4/2005

NCG93 NC-Gesellschaft: Werkstatt-Orientierte Produktionskonzepte. Empfehlung der NC-Gesellschaft. Ulm: Eigendruck, 1993

NCG05 NC-Gesellschaft: Ist DIN 66025 zukunftstauglich? EMO-Podiumsdiskussion CAD/CAM/NC-Verfahrenskette der Zukunft. Eigendruck NCG, Sept. 2005

NORD04 Nordmann: Messtechnische Lösungen für die Werkzeugüberwachung in Drehmaschinen. In: Drehteil + Drehmaschine, 2004, Nr. 2, S. 90-95

NORD04a Nordmann, K.: Prozessüberwachung beim Schleifen und Abrichten. In: Schleifen & Polieren, 2004, Nr. 5, S. 52-56

OPEN04 Open Mind: hyperMill hyperCAD – Zukunftsweisende CAD/CAM-Lösung. Unterföhring, 2004. Firmenschrift

OPTI97 ESPRIT project 8643 OPTIMAL – Optimised Preparation of Manufacturing Information with Multi-Level CAM-CNC Coupling. Final Report. Brüssel: Europäische Kommission, 1997

OSAC00 N.N.: OSACA Handbook 2.0. In: OSACA e.V. (Hrsg.). Eigendruck des OSACA e.V.; Stuttgart, 2000

PARA00 Parasuraman, R.; Sheridan, T. B.; Wickens, C. D.: A Model for Types and Levels of Human Interaction with Automation. In: IEEE Transactions on Systems, Man, and Cybernetics – Part A, Vol. 30, 2000, S. 286-297

PFEI06 Pfeiffer, T.; Schmitt, R.; et al.: Autonome Produktionszellen – Komplexe Produktionsprozesse flexibel automatisieren. Springer, Berlin 2006

PRIT93 Pritschow, G.: Open system controllers – a challenge for the future of the machine tool industry. Anals of the CIRP, Vol. 42/1, 1993, S. 449-452

PRIT03 Pritschow, G.; Weck, M.; Bauer, G.; Kahmen A.; Wild, M.: Offene, echtzeitfähige Steuerungsarchitektur – OCEAN: Open Controller Enabled by an Advaced Real-Time Network. In: wt Werkstattstechnik online. Jg. 93, 2003, Nr. 5, S. 374-378

PROD97 N.N.: NC-Steuerungen stark im Kommen. In: Produktion 17.4.1997 Nr. 16, S.9, 1997

PROM00 N.N.: PROMOS 2 / Real Time Monitor (RTM). In: Firmenprospekt der PROMETEC GmbH, 2000

PROM06 N.N.: Acfeed – Stückkosten und Bearbeitungszeit senken, Prozesssicherheit steigern. In: Firmenprospekt der PROMETEC GmbH, 2006

REHS98 Rehse, M.: Flexible Prozessüberwachung bei der Bohr und Fräsbearbeitung in einer Autonomen Produktionszelle. Dissertation RWTH Aachen, 1998

REIM04 Reimann, J.; Mey, H.: Offene Steuerung öffnet den japanischen Markt. In. NC – Fertigung. Nr. 1, 2004

RIEG94 Reiger, H.: Semantikorientierte Features zur kontinuierlichen Unterstützung der Produktgestaltung. Dissertation. Technische Universität Berlin. 1994

ROMM04 Rommel, B.; Bär, T.; Haasis, S.: 3D-Toleranzen für die Produktion. In: CAD-CAM Report Nr.3, 2004

SAUE04 Saueressig, F: 32 Bahnachsen voll im Griff. In: PC Control – The New Automation Technology Magazine. (Beckhoff) 2004, Nr. 2, S. 12-15

SAUR03 Saur, U.: Automatisch optimierte Zerspanung. In: MAV – Maschinen Anlagen Verfahren, Nr. 5 / 2003, S. 52 ff.

SCHE94 Schenk, D.; Wilson, P.: Information Modeling: The EXPRESS Way. Oxford University Press, 1994

SCHI95 Schindler, W.; Hartmann, E.; Fuchs-Frohnhofen, P.: Handlungsorientierte und facharbeitergerechte Gestaltung von Arbeitsabläufen. In: Arbeit, Nr. 2 / 1995

SCHI05 Schinerl, C.: Kollisionsvermeidung in die Maschinensteuerung integriert. In: MAV – Maschinen Anlagen Verfahren, Nr. 8 / 2005, S. 34 ff.

SCHL95 Schlick, C.; Kaever, M.: Facharbeiteranforderungen an Benutzungsschnittstellen von CNC-Werkzeugmaschinen. Institut für Arbeitswissenschaften, Aachen 1995

SCHL03 Schlenker, N.: Software ersetzt Hardware – Werkzeugüberwachung als reine Softwarelösung ohne Einsatz von zusätzlichen Sensoren. In: VDMA Nachrichten, Nr. 1, 2003

SCHU00 Schulze, H.: Erfahrungsgeleitete Arbeit als Leitbild für die Entwicklung und Gestaltung von Produktionssystemen in der industriellen Fertigung. Dissertation, Universität Hamburg, 2000.

SCHL02 Schulze-Lauen, H.: Kollisionsfreie Bahnerzeugung für das fünfachsige Fräsen auf Basis eines neuen Datenmodells mit Splines. Dissertation RWTH Aachen, Aachen: Shaker Verlag, 2002

SCHO00 Schulze, O.: Datenmodell für die NC-Programmierung. Dissertation RWTH Aachen, Aachen: Shaker Verlag, 2000

SERC02 SERCOS Interface. Die international genormte digitale Schnittstelle zur Kommunikation zwischen Steuerungen und Antrieben in numerisch gesteuerten Maschinen. Technische Kurzbeschreibung. Interessensgemeinschaft SERCOS Interface e.V. (Hrsg.), Stuttgart, 2002

SERC06 N.N.: Die 31.15µs Story – Harte Echtzeit. Interessensgemeinschaft SERCOS Interface e.V. (Hrsg.), Stuttgart, 2006

SERC06a Fritz, S.; Staudt, S.: SERCOS überwacht spanabhebende Abläufe. In: SERCOS insight, URL: www.sercos-insight.de [Stand September 2006]

SENS07 Sense 8: Dokumentation der 3D Grafiklibrary World Toolkit R 9, 2007

SHIN05 Shin, S.J.; Suh, S.H.; Stroud, I.: Reincarnation of G-code based part programs into STEP-NC for turning applications. In: Proceedings of 2005 CAD/CAM Conference, Kangwon, Korea, 2005, S. 155-164

SIEM00 Sinumerik 840D. OEM-package NCK, Software release 5, User's Manual, Edition 05/00. Siemens AG, Erlangen, 2000

SIEM02 Siemens AG: ShopMill – Bedienen/Programmieren Ausgabe 11/2002 – SINUMERIK 840D/840Di/810D. Erlangen, 2002. Firmenschrift

SIEM03 Siemens AG: Durchgängig offen bis in den Kern – Sinumerik Open Architecture. Erlangen, 2003. Firmenschrift

SIEM03a Siemens AG: Sinumerik 840D OEM-package NCK, Software release 5, User's Manual, Edition 04/03. Siemens AG, Erlangen, 2003

SIEM04 Siemens AG: Bedienungsanleitung Ausgabe 03/2004 – SINUMERIK 840D/840Di/810D HMI Advanced. Erlangen, 2003. Firmenschrift

SIEM04a Siemens AG: Sinumerik 810D/840D Werkzeug- und Formenbau, Handbuch. Erlangen, 2004. Firmenschrift

SIEM07 Kleinfeld, K.: Hauptversammlung der Siemens AG – Lagebericht des Hauptvorsitzenden. München, 25. Januar 2007

SÖHN03 Söhner, J.: Beitrag zur Simulation zerspanungstechnologischer Vorgänge mit Hilfe der Finite-Element-Methode. Dissertation. Universität Karlsruhe. Karlsruhe, 2003

SPUR04 Spur, G.: Über den Wandel der Produktionskultur. In: Klocke, F.; Pritschow, G. (Hrsg.): Autonome Produktion. Springer, Berlin, 2004

STEP99 ESPRIT project 20708 STEP-NC – STEP-Compliant Data Interface for Numerical Controls. Deliverable 1.1 – Demands of manufacturers and users towards the new interface. Europäische Kommission, Brüssel, 1999

STEP01 ESPRIT project 29708 STEP-NC – STEP-Compliant Data Interface for Numerical Controls. Final Report. Brüssel: Europäische Kommission, 2001

STEP04 GROWTH project GRD1-2001-00007 IMS STEP-NC – STEP-Compliant Data Interface for Numerical Controls. Final Report. Europäische Kommission, Brüssel, 2004

STEP06 Projektbeschreibung zu STEP-NC und AP 238. In: Website des Softwarehauses STEP Toos Inc. URL: www.steptools.com [Stand Dezember 2006]

SUH02 Suh, S.-H.; et al.: Architecture and implementation of a shop-floor programming system for STEP-compliant CNC. Computer Aided Design

SUH02a Suh, S.-H.; Cheon, S.: A framework for an intelligent CNC data model. In: International Journal of Advanced Manufacturing Technology, Juli 2002, Vol. 19(10), S. 729-735

TAYL07 Taylor, F.W.: On the Art of Cutting Metals. In: Transactions ASME 28, 1907, Nr. 1119, S. 31-350

TLUS92 Tlusty, J.; Smith, S.: Stabilizing Chatter by Automatic Spindle Speed Regulation. In: Anals of the CIRP. Vol. 41/1, 1992

TÖNS96 Tönshoff, H. K., et al.: Autonome, kooperative Produktionssysteme. VDI-Z 138 Nr. 9, 1996, S. 56 -59

TOEN00 Tönshoff, H. K.; Rackow, N.; Böß, V.: Virtual Reality in der NC-Programmierung. In: wt – Werkstatttechnik online. 2000, Nr. 7/8, S. 297-301

TOEN01 Tönshoff, H.-K.; Storr, A.; Selle, J.; Heusinger, S.: Präventive Fehlerkompensation beim Schlichtfräsen – Technologie- und steuerungsbasierte Fehlerkorrektur. In: wt - Werkstattechnik online. 2001, Nr. 7, S. 398-403

VANJ98 Vanja, S.: Durchgängige Produktmodellierung mit Features. In: CAD-CAM Report, 1998, Nr. 3

VDI2218 VDI-Richtlinie 2218: Feature-Technologie. Beuth Verlag, Berlin, 1999

VDI3426 Richtlinie VDI/VDE 3426 (1975). Numerisch gesteuerte Arbeitsmaschinen – Adaptive Control an spanenden Werkzeugmaschinen. Gesellschaft Meß- und Regelungstechnik (GMR)

VDIZ04 N.N.: Dienstleistungen rund um die Werkzeugmaschine. In: VDI-Z. 146, Nr. 9, 2004, S. 29

VDW02 Reinhart, G.; u.a.: Werkzeugmaschine 2010. Voruntersuchung zum Thema Werkzeugmaschine 2010. Studie des VDW/WGP. Karlsruhe, 2002

VDW04 VDW.: Deutsche Werkzeugmaschinen – Zahlen und Fakten. URL: www.vdw.de [Stand Dezember 2004]

VITR06 Vitr, M.: Steuerungstechnik (ab 1962). In: Eversheim, W.; Pfeiffer, T.; Weck, M. (Hrsg.): 100 Jahre Produktionstechnik – Werkzeugmaschinenlabor WZL der RWTH Aachen von 1906 bis 2006. Berlin: Springer, 2006, S. 315-351

VERS04 Versch, A.; Lange, D.: High Speed Cutting fest im Griff. In: Fertigung, 8/9 2004, S. 95 - 97

WECK77 Weck, M.; Teipel, K.: Dynamisches Verhalten spanender Werkzeugmaschinen. Einflussgrößen Beurteilungsverfahren Messtechnik. Springer, Berlin, 1977

WECK93a Weck, M., et al.: Die offene Steuerung – Zentraler Baustein leistungsfähiger Produktionsanlagen. In: Wettbewerbsfaktor Produktionstechnik. AWK Aachener Werkzeugmaschinenkolloquium ,93. Eversheim, W.; König, W.; Pfeifer, T.; Weck, M. (Hrsg.). VDI-Verlag, Düsseldorf, 1993

WECK93b Weck, M.; Kohring, A.; Klein, F.: Offene NC-Systeme, Grundlage herstellerunabhängiger Flexibilität. VDI-Z 135, Nr. 5, 1993, S. 51 - 55

WECK96 Weck, M., et al.: „Wie offen Hätten Sie's denn gern?" – Offene Systeme in der Fertigung. In: Wettbewerbsfaktor Produktionstechnik. AWK Aachener Werkzeugmaschinenkolloquium ‚96. Eversheim, W.; Klocke, F.; Pfeifer, T.; Weck, M. (Hrsg.). VDI-Verlag, Düsseldorf, 1996

WECK01 Weck, M.; Wolf, J.; Kiritsis, D.: STEP-NC – The STEP compliant NC Programming Interface: Evaluation and Improvement of the modern Interface, Int: IMS Forum. Ascona / Switzerland. 2001

WECK01a Weck, M.; Platen, S.; Kahmen, A.: Vereinigung von NC-Steuerung und Prozessüberwachung – Verknüpfen der Sensor- und Steuerungsinformationen eröffnet neue Potentiale. In: wt – Werkstattstechnik online. 2001, Nr. 5, S. 290 - 294

WECK01c Weck, M.: Werkzeugmaschinen Fertigungssysteme Band 3: Mechatronische Systeme, Vorschubantriebe, Prozessdiagnose. Springer, Düsseldorf, 2001

WECK01d Weck, M.: Werkzeugmaschinen Fertigungssysteme Band 4: Automatisierung von Maschinen und Anlagen. Springer Verlag, Düsseldorf, 2001

WECK01e Weck, M.: Werkzeugmaschinen Fertigungssysteme Band 5: Messtechnische Untersuchung und Beurteilung. Springer Verlag, Düsseldorf, 2001

WECK03 Weck, M.; Wolf, J.: ISO 14649 Provides Information for Sophisticated and Flexible Numerically Controlled Production. WGP, 2003

WECK03a Weck, M.; Wolf, J.: Der Datenfluss in der Fertigung von morgen - Mögliche Funktionen und deren Anforderungen. Fertigungstechnisches Kolloquium Stuttgart (FTK), 2003

WEIS07 Weiss, H.: DotGNU bereit .NET aus seiner Abhängigkeit. In: Elektronik. 2007, Nr. 3, S. 58-61

WENK06 Wenk, C.: Einsatz von Internet-Technologie für Maschinenbedienoberflächen. Dissertation RWTH Aachen, 2006

WEYD99 Weydandt, D.; Westerwick, A.; Henning, K.: Neue praxisnahe Möglichkeiten zur einer prospektiven, innovativen Technikfolgeabschätzung. In: Bröchler, S. et. al (Hrsg): Handbuch Technikfolgeabschätzung, Edition Sigma: Berlin, 1999, S. 263-278

WEYR00 Weyrich, N.; Rommel, B.; Haasis, S.; Müller, P.: First Prototype of a NC Controller based on STEP-NC. ProStep Science Days. URL: www. prostep.de/sciencedays2000/DaimlerChrysler_Weyrich.pdf [Stand 2000]

WOLF00 Wolf, J.; Hyun, Y.T.: CAM und NC sprechen dieselbe Sprache. STEP-NC, Vorteile der neuen Datenschnittstelle für die NC-Bearbeitung. In: Tools, Institutsschrift WZL und Fraunhofer IPT, RWTH Aachen, 2000, Nr. 1

WOLF01 Wolf, J.: STEP-NC – integrating shop floor into industrial dataflow for the enabling of intelligent near to process functions. PDT-Days Brüssel, 2001

11. Anhang

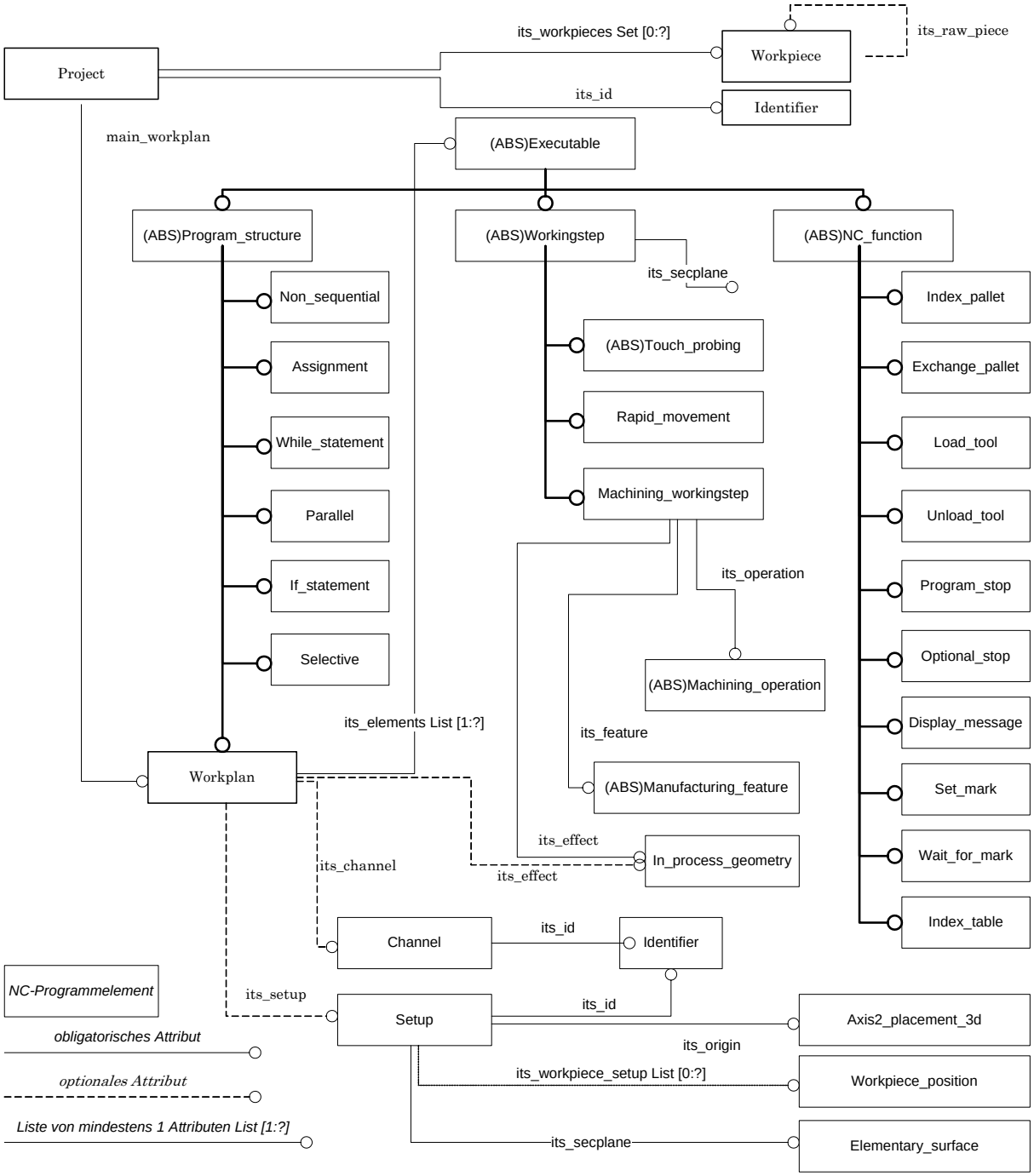
Appendix

11.1. Auszüge aus der entwickelten NC-Programmierschnittstelle ISO 14649

Extracts from the developed NC-programming interface ISO 14649

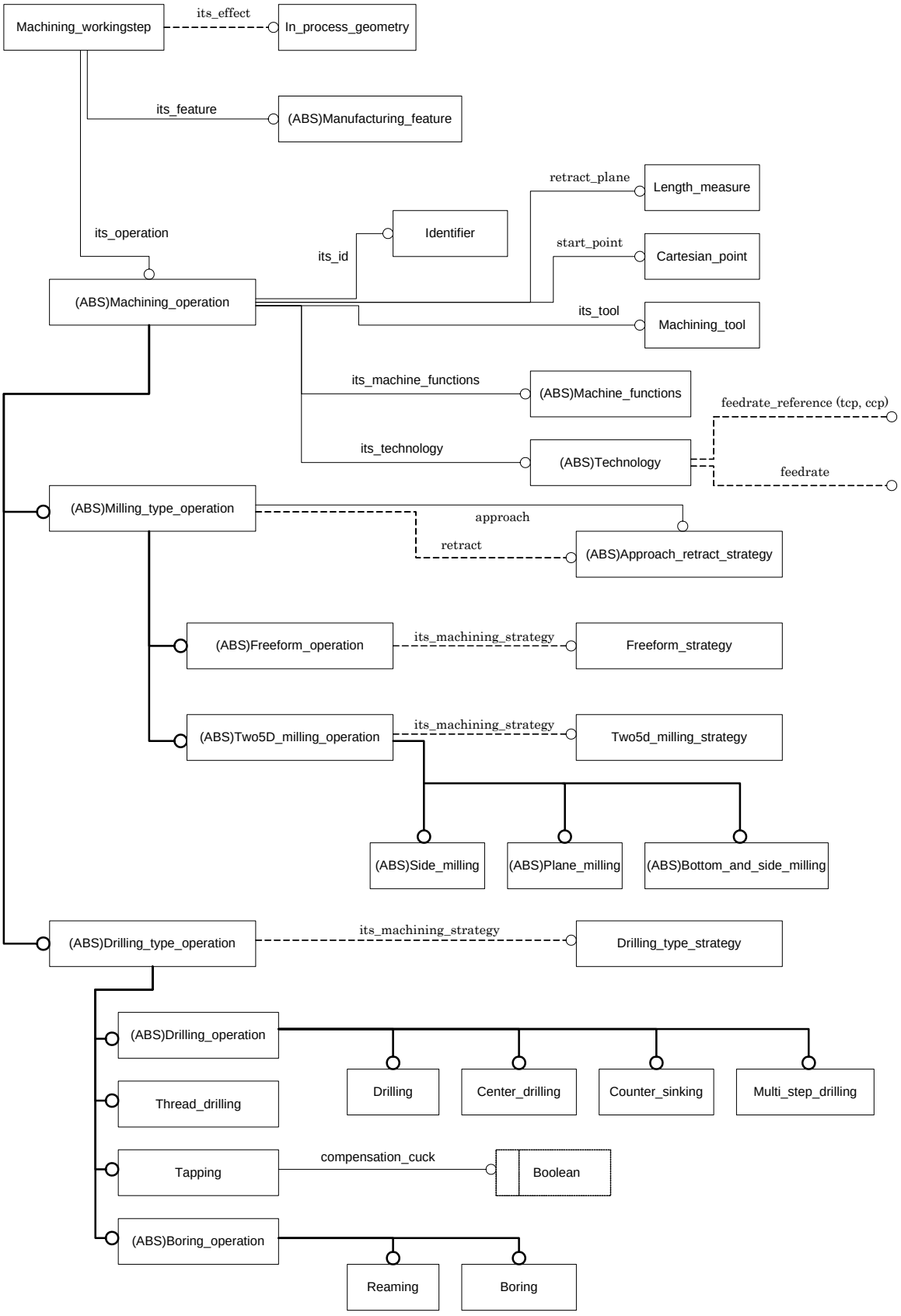
11.1.1. Projekt, Arbeitsplan, Arbeitsschritt und NC-Funktionen

Project, Workplan, Workingstep, NC-Function



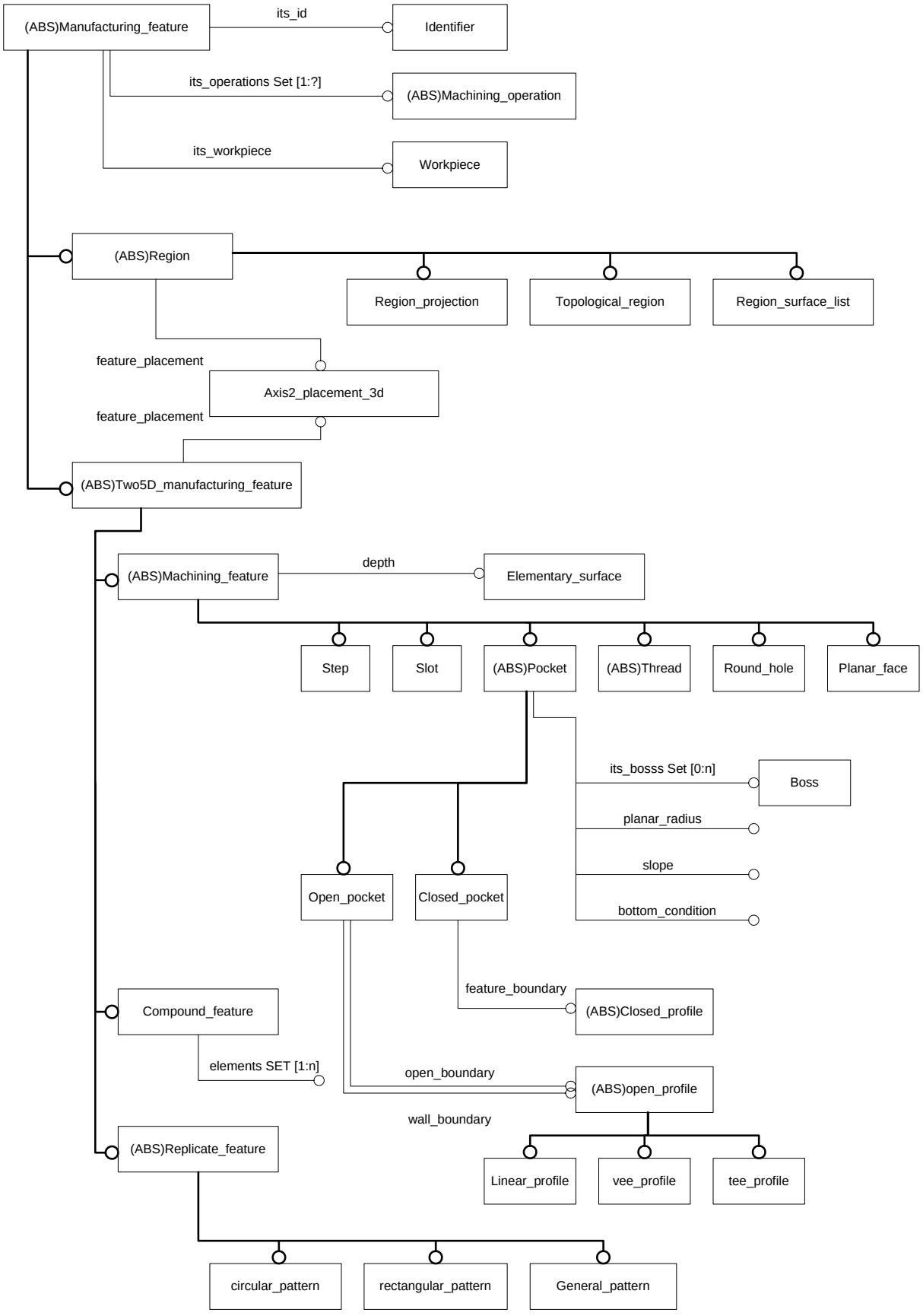
11.1.2. Arbeitsschritt zur Bearbeitung eines Features mit einer Operation

Workingstep defining which Feature to machine by which Operation



11.1.3. Geometrie- und Bearbeitungsmerkmale

Geometry- and machining Features



11.2. Beispielhafte Erstellung eines ISO 14649 NC-Programms

Example of how to generate an ISO 14649 NC-program

Gleichwohl in der Praxis STEP-NC Programme nicht händisches gelesen oder editiert werden sollten, ist nachfolgend zum besseren Verständnis ein Programm abgebildet mit Erläuterungen zu den verwendeten Datenelementen und dem kodierten Inhalt.

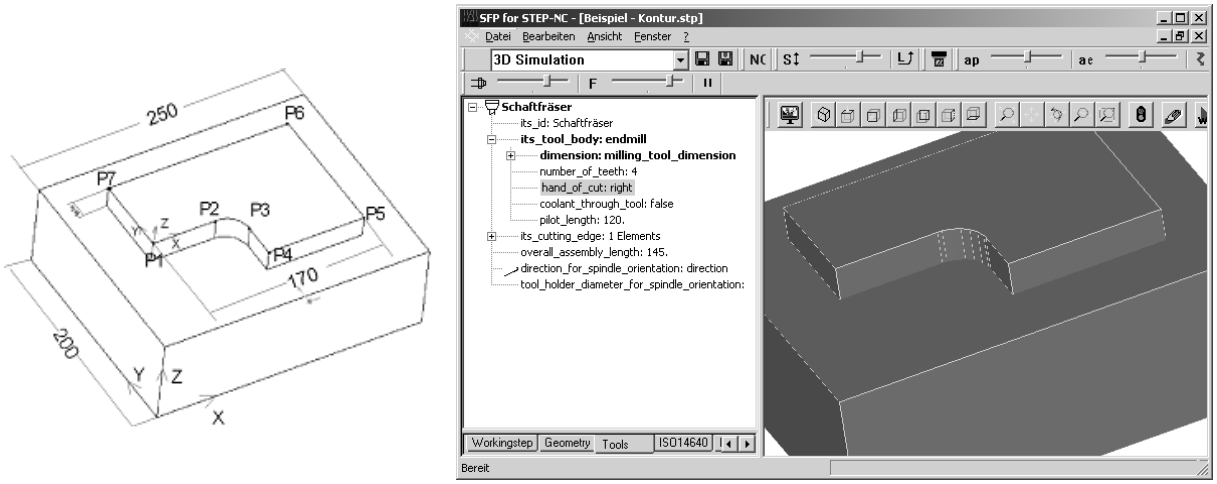


Bild 11.1: Fertigteil mit einer durch Fräsen herzustellenden Kontur

Finished part with a contour that shall be machined by milling

<u>Auszug aus dem Datenmodell der ISO 14649</u>

<u>Notation in EXPRESS ISO 10303-11</u>

```
ENTITY project;
 its_id:          identifier;
 main_workplan:   workplan;
 its_workpieces: SET [0:?]OF workpiece;
 its_owner: OPTIONAL person_and_address;
 its_release:     OPTIONAL date_and_time;
 its_status:      OPTIONAL approval;
END_ENTITY;
```

Das WOP-Tool generiert zu Beginn ein leeres NC-Programm, das nur einige wenige Elemente enthält:

Hierzu zählen das Projekt #1 mit einem Arbeitsplan #2 sowie den organisatorischen Daten und einem ersten Setup #3.

Nicht gesetzte Attribute erscheinen als „$“ oder bleiben leer.

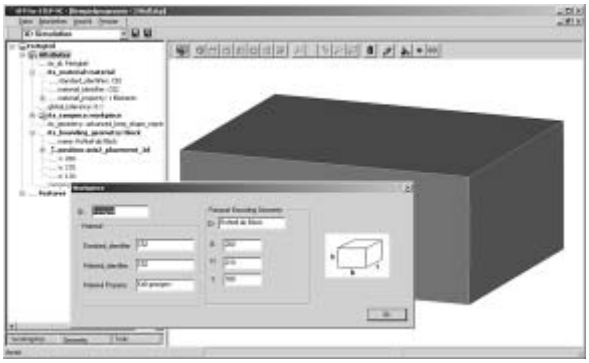
```
ISO-10303-21;
FILE_DESCRIPTION(('Beispiel'),'1');
FILE_NAME('Beispiel - Kontur.stp','2007-11-30T16-46-15',('WJ'),(''),'WZL parser package','','');
FILE_SCHEMA((' Milling schema - c0m0m1 - extended'));
ENDSEC;
DATA;
#1=PROJECT('Beispiel - Kontur',#2,(#10),#69,#4,#8);
#2=WORKPLAN('Arbeitsplan 1.Aufspannung',(#55),$,#3,$);
#3=SETUP('1. Aufspannung',#42,#46,(#51));
#4=DATE_AND_TIME(#5,#6);
#5=DATE(2008);
#6=LOCAL_TIME(15,38,53.0,#7);
#7=COORDINATED_UNIVERSAL_TIME_OFFSET(1,0,.AHEAD.);
#8=APPROVAL(#9,'draft');
#9=APPROVAL_STATUS('draft');
```

```
ENTITY workpiece;
 its_id:          identifier;
 its_material:    OPTIONAL material;
 global_tolerance:
      OPTIONAL shape_tolerance;
```

Wenn die Geometrie nicht aus dem CAD oder CAM übernommen wird, generiert der Programmierer dialoggeführt ein Fertigteil

```
 its_rawpiece:    OPTIONAL workpiece;
 its_geometry:    OPTIONAL
       avanced_brep_shape_representation;
 its_bounding_geometry:
       OPTIONAL bounding_geometry_select;
 clamping_positions:
       SET [0:?] OF cartesian_point;
END_ENTITY;

ENTITY block
 SUBTYPE OF
       (geometric_representation_item);
 name     : label;
 position : axis2_placement_3d;
 x        : positive_length_measure;
 y        : positive_length_measure;
 z        : positive_length_measure;
END_ENTITY;
```

#10 mit Materialeigenschaften #11 sowie Rohteil #12. Dazu kann er anstelle komplexer Flächen- und Volumenmodelle auf vereinfachte Geometrien, wie z.B. einen Block #13 zurückgreifen.



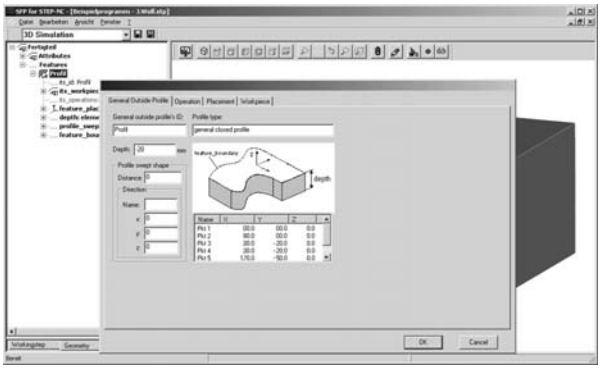
```
#10=WORKPIECE('Fertigteil',#11,0.01,#12,$,#13,());
#11=MATERIAL('C52','C52',());
#12=WORKPIECE('Rohteil',#11,0.50,$,$,#13,());
#13=BLOCK('Hilfsgeometrie für Rohteil',#14,250.0,200.0,30.0);
#14=AXIS2_PLACEMENT_3D('Koordinatensystem des Blocks',#15,#16,#17);
#15=CARTESIAN_POINT('Position Block zu Rohteil',(0.0,0.0,0.0));
#16=DIRECTION('z-Richtung Block',(0.0,0.0,1.0));
#17=DIRECTION('x-Richtung Block',(1.0,0.0,0.0));
```

```
ENTITY manufacturing_feature
 its_id :          identifier;
 its_workpiece :   workpiece;
 its_operations:
       SET [0:?] of machining_operation;
END_ENTITY;

ENTITY two5D_manufacturing_feature
 SUBTYPE OF (manufacturing_feature)
 feature_placement : axis2_placement_3d;
END_ENTITY;

ENTITY machining_feature
 SUBTYPE OF (two5D_manufacturing_feature)
 depth : elementary_surface;
END_ENTITY;

ENTITY profile_feature
 SUBTYPE OF (machining_feature)
 profile_swept_shape : linear_path;
END_ENTITY;

ENTITY general_outside_profile
 SUBTYPE OF (profile_feature)
 feature_boundary : profile;
END_ENTITY;
```

Ebenso können einzelne Feature dialogbasiert dem Werkstück hinzugefügt werden.

Hier wird ein Außenprofil #18 mit einer Kontur #19 und seiner Lage zum Werkstück #21 definiert. Die Kontur wird hier in Form einzelner Segmente #28, #29 mit Bögen #30 oder Polylinien #34 beschrieben.

An dieser Stelle greifen die Vererbungsmechanismen des EXPRESS-Schemas. Das „General_Outside_Profile“ erbt von den übergeordneten Elementen deren Attribute.

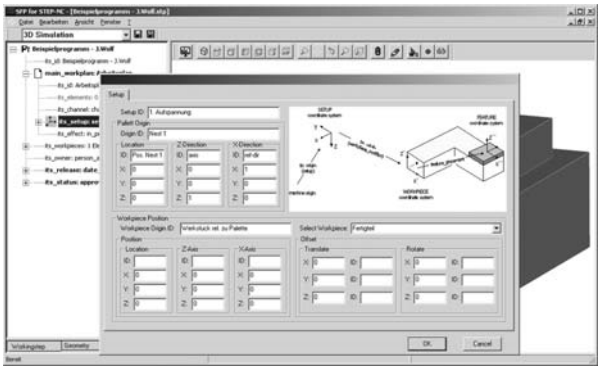


```
#18=GENERAL_OUTSIDE_PROFILE('Profil-Feature',#10,(#59),#20,#24,$,#19);
#19=GENERAL_CLOSED_PROFILE($,#27);
#20=AXIS2_PLACEMENT_3D('Position des Profils relativ zum Werkstück',#21,#22,#23);
#21=CARTESIAN_POINT('Position des Profils relativ zum Werkstück',(40.0,90.0,30.0));
#22=DIRECTION('z-Richtung des Profils',(0.0,0.0,1.0));
#23=DIRECTION('x-Richtung des Profils',(1.0,0.0,0.0));
#24=ELEMENTARY_SURFACE('Tiefe des Profils',#25);
```

```
#25=AXIS2_PLACEMENT_3D('Tiefe des Profils',#26,#22,#23);
#26=CARTESIAN_POINT('Tiefe des Profils',(0.0,0.0,-20.0));
#27=COMPOSITE_CURVE('Profil der Kontur',(#28,#29),.F.);
#28=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#34);
#29=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#30);
#30=TRIMMED_CURVE('',#31,(0.0),(90.0),.T.,.PARAMETER.);
#31=CIRCLE('Kreisbogen am Profil P2-P3',#32,20.0);
#32=AXIS2_PLACEMENT_3D('Kreismittelpunkt',#33,#22,#23);
#33=CARTESIAN_POINT('Kreismittelpunkt',(60.0,-20.0,0.0));
#34=POLYLINE('Polyline des Profils P3-P2',(#37,#38,#39,#40,#41,#35,#36,#37));
#35=CARTESIAN_POINT('P1',(0.0,0.0,0.0));
#36=CARTESIAN_POINT('P2',(60.0,0.0,0.0));
#37=CARTESIAN_POINT('P3',(80.0,-20.0,0.0));
#38=CARTESIAN_POINT('P4',(80.0,-50.0,0.0));
#39=CARTESIAN_POINT('P5',(170.0,-50.0,0.0));
#40=CARTESIAN_POINT('P6',(170.0,70.0,0.0));
#41=CARTESIAN_POINT('P7',(0.0,70.0,0.0));
```

```
ENTITY setup
 its_id :          identifier;
 its_origin:       OPTIONAL
       axis2_placement_3d;
 its_secplane:     elementary_surface;
 its_workpiece_setup:
       LIST [0:?] OF workpiece_setup;
END_ENTITY;

ENTITY workpiece_setup
 its_workpiece :   workpiece;
 its_origin :      axis2_placement_3D;
 its_offset :      OPTIONAL offset_vector;
 its_restricted_area :
       restricted_area_select;
 its_instructions :
       LIST [0:?] OF setup_instructions;
END_ENTITY;
```

Bevor die Bearbeitung der Kontur geplant wird, sollte im Arbeitsplan die Spannlage #3 definiert werden.

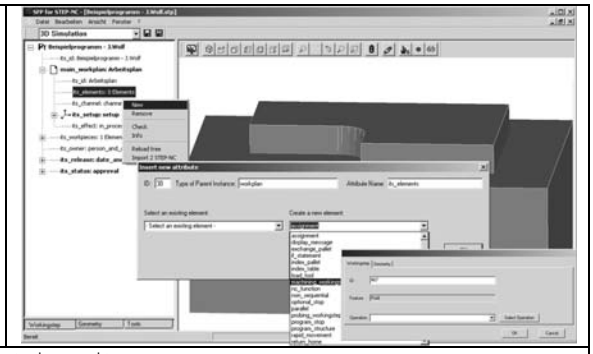


```
#3=SETUP('1. Aufspannung',#42,#46,(#51));
...
#42=AXIS2_PLACEMENT_3D('1.Aufspannung rel. zur Maschine',#43,#44,#45);
#43=CARTESIAN_POINT('relativ zur Maschine',(12.0,13.0,1.40));
#44=DIRECTION('axis',(0.0,0.0,1.0));
#45=DIRECTION('ref-dir',(0.0,1.0,0.0));
#46=ELEMENTARY_SURFACE('Globale Sicherheitsebene des Arbeitsplans',#47);
#47=AXIS2_PLACEMENT_3D('Gl. Sicherheitsebene des Arbeitsplans',#48,#49,#50);
#48=CARTESIAN_POINT('location',(0.0,0.0,50.0));
#49=DIRECTION('axis',(0.0,0.0,1.0));
#50=DIRECTION('ref-dir',(1.0,0.0,0.0));
#51=WORKPIECE_SETUP(#12,#52,$,$,(#54));
#52=AXIS2_PLACEMENT_3D('Rohteil relativ zur Aufspannung',#53,#49,#50);
#53=CARTESIAN_POINT('Rohteil relativ zur Aufspannung',(1.0,2.0,3.0));
#54=SETUP_INSTRUCTION('Bauteil mit Spannelementen fixieren',$);
```

```
ENTITY executable
 its_id :       identifier;
END_ENTITY;

ENTITY workingstep
 SUBTYPE OF (executable);
 its_secplane: elementary_surface;
END_ENTITY;

ENTITY machining_workingstep
 SUBTYPE OF (workingstep);
 its_secplane: elementary_surface;
 its_feature:  manufacturing_feature;
 its_operation: machining_operation;
```

In den bestehenden Arbeitsplan #2 wird ein Arbeitsschritt #55 eingefügt. Dieser definiert an welchem Feature #18 welche Operation #59 abgearbeitet werden soll.

```
  its_effect: OPTIONAL in_process_geometry;
END_ENTITY;
```



```
#2=WORKPLAN('Arbeitsplan 1.Aufspannung',(#55),$,#3,$);
...
#55=MACHINING_WORKINGSTEP('Kontur schruppen',#56,#18,#59,$);
#56=ELEMENTARY_SURFACE('Sicherheitsebene des Arbeitsschritts',#57);
#57=AXIS2_PLACEMENT_3D('Sicherheitsebene des Arbeitsschritts',#58,#22,#23);
#58=CARTESIAN_POINT('Sicherheitsebene des Arbeitsschritts',(0.0,0.0,30.0));
#59=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Schruppoperation', …
```

```
ENTITY operation
 its_toolpath: OPTIONAL toolpath_list;
 its_tool_direction: OPTIONAL tool_dir…;
END_ENTITY;


ENTITY machining_operation
 SUBTYPE OF (operation);
 its_id:          identifier;
 retract_plane:   length_measure;
 start_point:     OPTIONAL
       cartesian_point;
 its_tool:        machining_tool;
 its_technology: technology;
 its_machine_functions:
       machine_functions;
END_ENTITY;

...
```

Die Operation #59 wird durch eine Vielzahl von Parametern beschrieben. Insbesondere gehören hierzu die Strategie #60, Technologie #65, Maschinenfunktionen #68 sowie das Werkzeug #61.

Die Elemente Ext_Milling_Technology #65 und Ext_Adaptive_Control #66 wurden von den standardisierten Elementen Milling_Technology und Adaptive_Control abgeleitet. Sie enthalten zusätzliche Grenzwerte für die realisierten Demoszenarien.

```
#59=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Schruppoperation',20.0,$,#61,#65,#68,25.0,
      $,$,#60,25.0,7.0,0.75,1.25);
#60=CONTOUR_PARALLEL(5.0,.F.,.CW.,.CONVENTIONAL.);
...
#65=EXT_MILLING_TECHNOLOGY($,.TCP.,3.34,5000.0,0.50,.F.,.F.,.F.,#66, ' Grenzen ae',
      (2.00,12.0), ' Grenzen S' ,(1000.0,5000.0), ' Grenzen f' ,(0.05,0.75));
#66=EXT_ADAPTIVE_CONTROL('zulässige Bandbreite Spindelstrom in%', (-0.5,0.5));
...
#68=MILLING_MACHINE_FUNCTIONS(.F.,80.0,$,.F.,$,(),.F.,$,$,());
#61=MILLING_CUTTING_TOOL('Schaftfräser',#62,(#63),145.0,$,$);
```

```
ENTITY machining_tool
 its_id :         identifier;
END_ENTITY;

ENTITY cutting_tool
 ABSTRACT SUPERTYPE
 SUBTYPE OF (machining_tool);
 its_tool_body:   tool_body;
 its_cutting_edge:
      LIST [1:?] OF cutting_component;
overall_assembly_length:
      OPTIONAL length_measure;
END_ENTITY;

ENTITY milling_cutting_tool
 SUBTYPE OF (cutting_tool);
 direction_for_spindle_orientation : …;
 tool_holder_diameter_for_spindle_orienta
 tion : OPTIONAL length_measure;
END_ENTITY;
```

Das Werkzeug #61 wird je nach Typ unterschiedlich parametriert. Dabei kann jede einzelne Schneide #63 in Form von Geometrie und Material #64 beschrieben werden.

Technologische Grenzen des Werkzeugs bildet das entwickelte und standardisierte ISO 14649 Datenmodell z.Zt. nicht ab.

```
#61=MILLING_CUTTING_TOOL('Schaftfräser',#62,(#63),145.0,$,$);
```

```
#62=ENDMILL(#67,4,$,.F.,120.0);
#63=CUTTING_COMPONENT(80.0,#64,$,350.0,#65);
#64=MATERIAL('HSS','HSS',());
. . .
#67=MILLING_TOOL_DIMENSION(30.0,$,$,$,0.40,$,$);
#69=PERSON_AND_ADDRESS(#70,$);
#70=PERSON('WJ','Wolf','Jochen',(),(),());
ENDSEC;
END-ISO-10303-21;
```

11.3. ISO 14649 Beispielprogramm des STEP-NC Projektkonsortiums

ISO 14649 sample NC-program generated and machined within the STEP-NC project

Im Folgenden ist ein mit den realisierten Systemen geplantes, kodiertes und abgearbeitetes NC-Programm abgedruckt. Es enthält die vollständige Aufgaben- und Geometriebeschreibung zur Fertigung des in den Bild 11.2 dargestellten Bauteils.

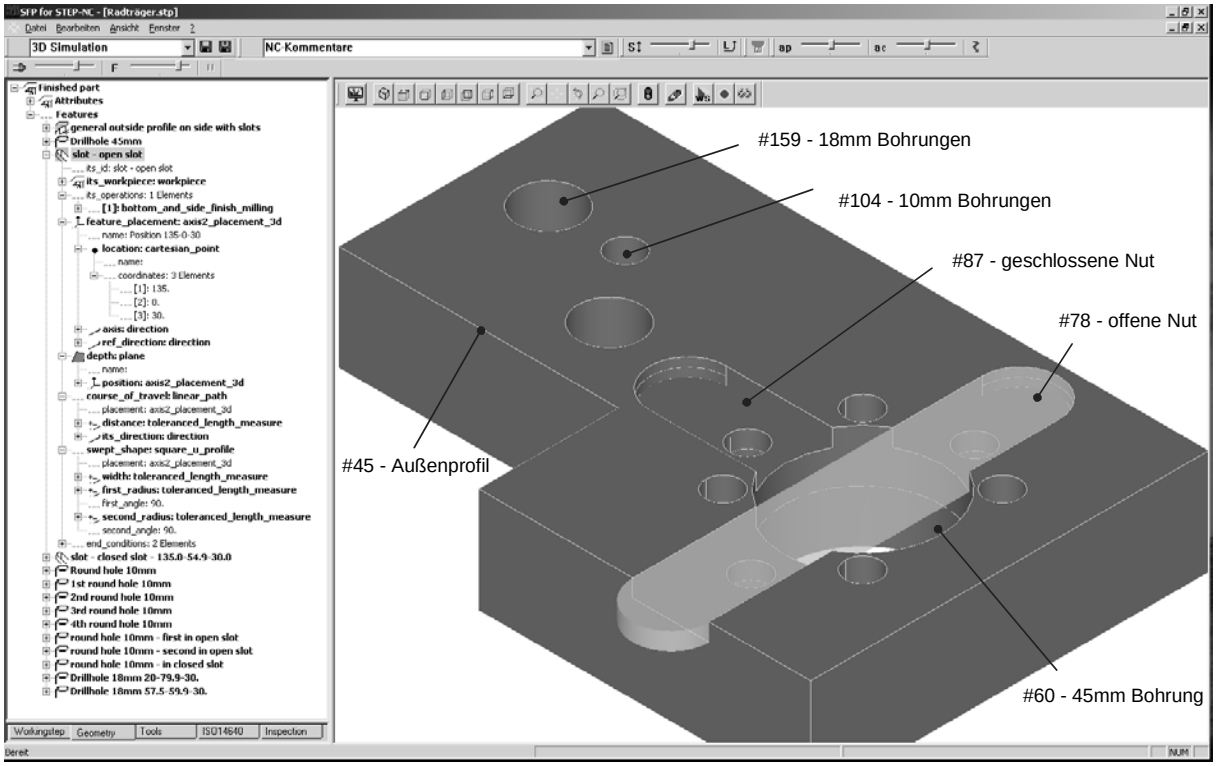


Bild 11.2: Ansicht des Referenzwerkstücks im realisierten WOP-System
Reference part generated and programmed in the realized SFP-System

Zur einfacheren Lesbarkeit sind die Referenzen ausgehend von den Arbeitsschritten sortiert worden. Die wichtigsten Elemente des NC-Programms finden sich unter folgenden Referenzen:

Projekt	#186
Arbeitsplan	#180, #179
Arbeitsschritte	#46,#61,#79,#88,#105,#111,#116,#121,#126,#137,#141,#145,#160,#165
Merkmale	#45,#60,#78,#87,#104,#110,#115,#120,#125,#134,#140,#144,#159,#164
Operationen	#26,#51, 2x #63, 5x #95, 3x #136, 2x #152

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('Referenzwerkstück der STEP-NC Konsortien für das Fräsen'),'1');
FILE_NAME('Radträger.stp','2006-4-6T11-54-7',('WJ'),(''),'WZL ISO10303-Part21 Bibliothek','','');
FILE_SCHEMA(('Milling schema - c0m0m1 - extended'));
ENDSEC;
DATA;
#1=CARTESIAN_POINT('',(0.0000,0.0000,30.0000));
#2=DIRECTION('z-Direction 0 0 +1',(0.0000,0.0000,1.0000));
#3=DIRECTION('',(1.0000,0.0000,0.0000));
#4=AXIS2_PLACEMENT_3D('',#1,#2,#3);
#5=PLANE('Security plane',#4);
#6=PROPERTY_PARAMETER('');
#7=MATERIAL('','',(#6));
#8=CARTESIAN_POINT('',(-2.5000,-2.5000,-2.5000));
#9=DIRECTION('',(0.0000,0.0000,1.0000));
#10=DIRECTION('',(1.0000,0.0000,0.0000));
#11=AXIS2_PLACEMENT_3D('',#8,#9,#10);
#12=BLOCK('',#11,180.0000,118.9000,35.0000);
#13=WORKPIECE('Stock',#7,$,$,$,#12,());
#14=CARTESIAN_POINT('',(0.0000,0.0000,0.0000));
#15=DIRECTION('',(0.0000,0.0000,1.0000));
#16=DIRECTION('',(1.0000,0.0000,0.0000));
#17=AXIS2_PLACEMENT_3D('',#14,#15,#16);
#18=BLOCK('',#17,175.0000,113.9000,30.0000);
#19=WORKPIECE('Finished part',#7,$,#13,$,#18,());
#20=MILLING_TOOL_DIMENSION(21.9000,$,$,118.9400,$,$,$);
#21=ENDMILL(#20,3,$,$,118.9400);
#22=MILLING_CUTTING_TOOL('Mill_22',#21,(),118.9400,$,$);
#23=MILLING_TECHNOLOGY($,$,$,100.0000,0.1000,$,$,$,$);
#24=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#25=UNIDIRECTIONAL(0.5000,$,$,$);
#26=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'',10.0,$,#22,#23,#24,10,$,$,#25,5.0,10,0.,0);
#27=CARTESIAN_POINT('',(0.0000,0.0000,30.0000));
#28=DIRECTION('',(0.0000,0.0000,1.0000));
#29=DIRECTION('',(1.0000,0.0000,0.0000));
#30=AXIS2_PLACEMENT_3D('',#27,#28,#29);
#31=CARTESIAN_POINT('',(0.0000,0.0000,-30.0000));
#32=DIRECTION('z-Direction 0 0 1',(0.0000,0.0000,1.0000));
#33=DIRECTION('x-Direction 1 0 0',(1.0000,0.0000,0.0000));
#34=AXIS2_PLACEMENT_3D('',#31,#32,#33);
#35=PLANE('Depth of general outside profile - side with slots',#34);
#36=CARTESIAN_POINT('',(0.0000,113.9000,0.0000));
#37=CARTESIAN_POINT('',(0.0000,40.0000,0.0000));
#38=CARTESIAN_POINT('',(80.0000,40.0000,0.0000));
#39=CARTESIAN_POINT('',(80.0000,0.0000,0.0000));
#40=CARTESIAN_POINT('',(175.0000,0.0000,0.0000));
#41=CARTESIAN_POINT('',(175.0000,113.9000,0.0000));
#42=CARTESIAN_POINT('',(0.0000,113.9000,0.0000));
#43=POLYLINE('',(#36,#37,#38,#39,#40,#41,#42));
#44=GENERAL_CLOSED_PROFILE($,#43);
#45=GENERAL_OUTSIDE_PROFILE('Außenprofil',#19,(#26),#30,#35,$,#44);
#46=MACHINING_WORKINGSTEP('its_id: Außenprofil - Konturfräsen',#5,#45,#26,$);
#47=CARTESIAN_POINT('',(0.0000,0.0000,5.0000));
#48=MILLING_CUTTING_TOOL('Mill_20',#21,(),118.9400,$,$);
#49=MILLING_TECHNOLOGY($,$,$,100.0000,0.1000,$,$,$,$);
```

```
#50=UNIDIRECTIONAL(0.5000,$,$,$);
#51=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Zirkularfräsen',10,#47,#48,#49,#24,$,$,$,#5
    0,5.0000,10.0000,0.0000,0.0000);
#52=CARTESIAN_POINT('',(135.0000,54.9000,30.0000));
#53=AXIS2_PLACEMENT_3D('',#52,#28,#29);
#54=CARTESIAN_POINT('',(0.0000,0.0000,-30.0000));
#55=AXIS2_PLACEMENT_3D('',#54,#2,#3);
#56=PLANE('',#55);
#57=PLUS_MINUS_VALUE(45.1000,44.9000,4);
#58=TOLERANCED_LENGTH_MEASURE(45.0000,#57);
#59=THROUGH_BOTTOM_CONDITION();
#60=ROUND_HOLE('Drillhole 45mm',#19,(#51),#53,#56,#58,$,#59);
#61=MACHINING_WORKINGSTEP('45mm Bohrung - zirkular ausfräsen',#5,#60,#51,$);
#62=MILLING_TECHNOLOGY($,$,$,100.0000,0.1000,$,$,$,$);
#63=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'',10.0000,#47,#48,#62,#24,$,$,$,#50,2.5000,
    $,0.0000,0.0000);
#64=CARTESIAN_POINT('',(135.0000,0.0000,30.0000));
#65=AXIS2_PLACEMENT_3D('Position 135-0-30',#64,#28,#3);
#66=CARTESIAN_POINT('',(0.0000,0.0000,-5.0000));
#67=AXIS2_PLACEMENT_3D('',#66,#28,#3);
#68=PLANE('',#67);
#69=TOLERANCED_LENGTH_MEASURE(100.0000,$);
#70=DIRECTION('',(0.0000,1.0000,0.0000));
#71=LINEAR_PATH($,#69,#70);
#72=PLUS_MINUS_VALUE(22.0000,19.5000,4);
#73=TOLERANCED_LENGTH_MEASURE(21.9000,#72);
#74=TOLERANCED_LENGTH_MEASURE(0.2000,$);
#75=TOLERANCED_LENGTH_MEASURE(0.2000,$);
#76=SQUARE_U_PROFILE($,#73,#74,90.0000,#75,90.0000);
#77=RADIUSED_SLOT_END_TYPE();
#78=SLOT('slot - open slot',#19,(#63),#65,#68,#71,#76,(#77,#77));
#79=MACHINING_WORKINGSTEP('Offene Nut - linear fräsen',#5,#78,#63,$);
#80=CARTESIAN_POINT('',(135.0000,54.9000,30.0000));
#81=DIRECTION('',(0.0000,-1.0000,0.0000));
#82=AXIS2_PLACEMENT_3D('Position 135-54-30',#80,#28,#81);
#83=PLUS_MINUS_VALUE(50.1000,49.9000,3);
#84=TOLERANCED_LENGTH_MEASURE(50.0000,#83);
#85=DIRECTION('',(0.0000,-1.0000,0.0000));
#86=LINEAR_PATH($,#84,#85);
#87=SLOT('slot - closed slot - 135.0-54.9-30.0',#19,(#63),#82,#68,#86,#76,(#77,#77));
#88=MACHINING_WORKINGSTEP('Geschlossene Nut - linear fräsen',#5,#87,#63,$);
#89=CARTESIAN_POINT('',(0.0000,0.0000,1.0000));
#90=MILLING_TOOL_DIMENSION(9.9000,$,$,130.5000,$,$,$);
#91=ENDMILL(#90,$,$,$,130.5000);
#92=MILLING_CUTTING_TOOL('DRILL_10',#91,(),130.5000,$,$);
#93=MILLING_TECHNOLOGY(0.0100,.TCP.,$,100.0000,0.0001,.F.,.F.,.F.,$);
#94=DRILLING_TYPE_STRATEGY(1.0000,1.0000,0.0000,$,$,15.0000);
#95=DRILLING($,$,'Operation to drill 10mm drill-
    holes',5.0000,#89,#92,#93,#24,$,30.0000,0.0000,0.0000,$,#94);
#96=CARTESIAN_POINT('',(42.0000,79.9000,30.0000));
#97=AXIS2_PLACEMENT_3D('Position 42-79-30',#96,#28,#3);
#98=CARTESIAN_POINT('',(0.0000,0.0000,-30.0000));
#99=AXIS2_PLACEMENT_3D('',#98,#2,#3);
#100=PLANE('Depth of 18mm round hole',#99);
#101=PLUS_MINUS_VALUE(10.1000,9.5000,4);
#102=TOLERANCED_LENGTH_MEASURE(9.9000,#101);
#103=THROUGH_BOTTOM_CONDITION();
```

```
#104=ROUND_HOLE('Round hole 10mm',#19,(#95),#97,#100,#102,$,#103);
#105=MACHINING_WORKINGSTEP('10mm Bohrung - Bohren',#5,#104,#95,$);
#106=CARTESIAN_POINT('',(115.0000,34.9000,30.0000));
#107=AXIS2_PLACEMENT_3D('Position 115-34-30',#106,#28,#3);
#108=TOLERANCED_LENGTH_MEASURE(9.9000,#101);
#109=THROUGH_BOTTOM_CONDITION();
#110=ROUND_HOLE('1st round hole 10mm',#19,(#95),#107,#100,#108,$,#109);
#111=MACHINING_WORKINGSTEP('its_id: 10mm Bohrung c - bohren',#5,#110,#95,$);
#112=CARTESIAN_POINT('',(115.0000,74.9000,30.0000));
#113=AXIS2_PLACEMENT_3D('Position 115-74-30',#112,#28,#3);
#114=THROUGH_BOTTOM_CONDITION();
#115=ROUND_HOLE('2nd round hole 10mm',#19,(#95),#113,#100,#108,$,#114);
#116=MACHINING_WORKINGSTEP('its_id: 10mm Bohrung d - bohren',#5,#115,#95,$);
#117=CARTESIAN_POINT('',(155.0000,74.9000,30.0000));
#118=AXIS2_PLACEMENT_3D('Position 155-74-30',#117,#28,#3);
#119=THROUGH_BOTTOM_CONDITION();
#120=ROUND_HOLE('3rd round hole 10mm',#19,(#95),#118,#100,#108,$,#119);
#121=MACHINING_WORKINGSTEP('its_id: 10mm Bohrung a - bohren',#5,#120,#95,$);
#122=CARTESIAN_POINT('',(155.0000,34.9000,30.0000));
#123=AXIS2_PLACEMENT_3D('Position 155-34-30',#122,#28,#3);
#124=THROUGH_BOTTOM_CONDITION();
#125=ROUND_HOLE('4th round hole 10mm',#19,(#95),#123,#100,#108,$,#124);
#126=MACHINING_WORKINGSTEP('10mm Bohrung b - bohren',#5,#125,#95,$);
#127=CARTESIAN_POINT('',(135.0000,22.9000,30.0000));
#128=AXIS2_PLACEMENT_3D('Position 135-22-30',#127,#28,#3);
#129=CARTESIAN_POINT('',(0.0000,0.0000,15.0000));
#130=AXIS2_PLACEMENT_3D('',#129,#2,#3);
#131=PLANE('',#130);
#132=TOLERANCED_LENGTH_MEASURE(9.9000,$);
#133=CONICAL_HOLE_BOTTOM(60.0000,$);
#134=ROUND_HOLE('round hole 10mm - first in open slot',#19,(#95),#128,#131,#132,$,#133);
#135=CARTESIAN_POINT('',(0.0000,0.0000,1.0000));
#136=DRILLING($,$,'Drilling operation for drillholes in
    slots',5.0000,#135,#92,#93,$,$,15.0000,0.0000,0.0000,$,#94);
#137=MACHINING_WORKINGSTEP('10mm Bohrung in offener Nut - bohren',#5,#134,#136,$);
#138=CARTESIAN_POINT('',(135.0000,86.9000,30.0000));
#139=AXIS2_PLACEMENT_3D('Position 135-86-30',#138,#28,#3);
#140=ROUND_HOLE('round hole 10mm - second in open
    slot',#19,(#95),#139,#131,#132,$,#133);
#141=MACHINING_WORKINGSTEP('10mm Bohrung in offener Nut - bohren',#5,#140,#136,$);
#142=CARTESIAN_POINT('',(102.0000,54.9000,30.0000));
#143=AXIS2_PLACEMENT_3D('Position 102-54-30',#142,#28,#3);
#144=ROUND_HOLE('round hole 10mm - in closed slot',#19,(#95),#143,#131,#132,$,#133);
#145=MACHINING_WORKINGSTEP('10mm Bohrung in Nut - bohren',#131,#144,#136,$);
#146=CARTESIAN_POINT('',(0.0000,0.0000,2.0000));
#147=MILLING_TOOL_DIMENSION(17.9000,$,$,108.8900,$,$,$);
#148=ENDMILL(#147,$,$,$,108.8900);
#149=MILLING_CUTTING_TOOL('DRILL_18',#148,(),108.8900,$,$);
#150=MILLING_TECHNOLOGY(0.0100,.TCP.,$,100.0000,0.1000,$,$,$,$);
#151=MILLING_MACHINE_FUNCTIONS(.T.,$,$,$,$,(),$,$,$,());
#152=DRILLING($,$,'Drilling operation for 18mm drill-
    holes',5.0000,#146,#149,#150,#151,$,15.0000,0.0000,0.0000,$,#94);
#153=CARTESIAN_POINT('',(20.0000,79.9000,30.0000));
#154=DIRECTION('',(0.0000,0.0000,1.0000));
#155=DIRECTION('',(1.0000,0.0000,0.0000));
#156=AXIS2_PLACEMENT_3D('Position 20-79-30',#153,#154,#155);
#157=TOLERANCED_LENGTH_MEASURE(17.9000,$);
```

```
#158=THROUGH_BOTTOM_CONDITION();
#159=ROUND_HOLE('Drillhole 18mm 20-79.9-30.',#19,(#152),#156,#100,#157,$,#158);
#160=MACHINING_WORKINGSTEP('18mm Bohrung - bohren',#5,#159,#152,$);
#161=CARTESIAN_POINT('',(57.5000,59.9000,30.0000));
#162=AXIS2_PLACEMENT_3D('Position 57-59-30',#161,#28,#3);
#163=THROUGH_BOTTOM_CONDITION();
#164=ROUND_HOLE('Drillhole 18mm 57.5-59.9-30.',#19,(#152),#162,#100,#157,$,#163);
#165=MACHINING_WORKINGSTEP('18mm Bohrung - bohren',#5,#164,#152,$);
#166=CARTESIAN_POINT('',(-60.0000,80.0000,263.0000));
#167=DIRECTION('',(0.0000,0.0000,1.0000));
#168=DIRECTION('',(0.0000,1.0000,0.0000));
#169=AXIS2_PLACEMENT_3D('',#166,#167,#168);
#170=CARTESIAN_POINT('location',(0.0000,0.0000,10.0000));
#171=AXIS2_PLACEMENT_3D('',#170,#2,#3);
#172=PLANE('Global security plane',#171);
#173=CARTESIAN_POINT('',(0.0000,0.0000,0.0000));
#174=DIRECTION('',(0.0000,0.0000,1.0000));
#175=DIRECTION('',(1.0000,0.0000,0.0000));
#176=AXIS2_PLACEMENT_3D('',#173,#174,#175);
#177=WORKPIECE_SETUP(#19,#176,$,$,());
#178=SETUP('Setup for machining the side with slots',#169,#172,(#177));
#179=WORKPLAN('1. Aufspannung',(#46,#61,#79,#88,#105,#111,#116,
        #121,#126,#137,#141,#145,#160,#165), $,#178,$);
#180=WORKPLAN('Gesamtarbeitsplan',(#179),$,$,$);
#181=PERSON('','',(),(),());
#182=PERSON_AND_ADDRESS(#181,$);
#183=CALENDAR_DATE(2004,10,4);
#184=LOCAL_TIME(10,15,$,$);
#185=DATE_AND_TIME(#183,#184);
#186=PROJECT('Bearbeitungsprogramm - Radträger',#180,(#19),#182,#185,$);
ENDSEC;
END-ISO-10303-21;
```

Lebenslauf

Persönliche Angaben

Name, Vorname:	Wolf, Jochen
Geburtsdatum:	22. November 1973
Geburtsort:	Bergheim (Erft)
Familienstand:	verheiratet, zwei Kinder
Staatsangehörigkeit:	deutsch

Schulbildung

1979 – 1980	Preparatory School, Alexandria, USA
1980 – 1984	Grundschule, Buchholz
1984 – 1990	Kant-Gymnasium, Boppard
1990 – 1993	Konrad-Adenauer-Gymnasium, Meckenheim

Studium

1993 – 1999	Maschinenbau an der RWTH Aachen
1998 – 1999	Auslandsstudium an der Universidade Federal de Santa Catarina, Florianópolis, Brasilien

Wissenschaftlicher Werdegang

11/1995 – 09/1998	Studentische Hilfskraft am Fraunhofer Institut für Produktionstechnologie in Aachen
09/1998 – 03/1999	Studentische Hilfskraft am Laboratório de Mecânica de Precisão der UFSC in Florianópolis, Brasilien
09/1999 – 08/2004	Wissenschaftlicher Mitarbeiter am Werkzeugmaschinenlabor (WZL), Lehrstuhl für Werkzeugmaschinen der RWTH Aachen
09/2001 – 08/2004	Gruppenleiter NC-Technik am WZL

Weiterer beruflicher Werdegang

09/2004 – 09/2006	Leiter Entwicklung Nockenwellen, Muhr und Bender KG
10/2006 – 08/2008	Leiter Produktbereiche Schmiedenocken und Nockenwellen, Muhr und Bender KG, Mubea Motorkomponenten GmbH
seit 09/2008	Leiter Steuerungstechnik, Alfred H. Schütte GmbH & Co KG